

BENÍCIO JOSÉ DE SOUZA

"SOFTWARE DE UM MINICOMPUTADOR: SISTEMA BÁSICO DE CONTROLE"

"Dissertação de Mestrado" apresentada à Escola Politécnica da Universidade de São Paulo, para a obtenção do título de Mestre em Engenharia".

Área de Concentração - Engenharia de Eletricidade.

ORIENTADOR: Prof. Dr. Tamio Shimizu

FD-147  
e.2

São Paulo  
-1976-

BENÍCIO JOSÉ DE SOUZA

"SOFTWARE DE UM MINICOMPUTADOR: SISTEMA BÁSICO DE CONTROLE"

"Dissertação de Mestrado" apresentada à Escola Politécnica da Universidade de São Paulo, para a obtenção do título de Mestre em Engenharia".

Área de Concentração - Engenharia de Eletricidade.

ORIENTADOR: Prof. Dr. Tamio Shimizu

São Paulo

-1976-

DEDALUS - Acervo - EPEL



31500011875

A minha esposa,



## A G R A D E C I M E N T O S

Ao Prof. Dr. Tamio Shimizu pela orientação na realização deste trabalho;

Ao Prof. Dr. Antonio Marcos de Aguiar Massola pelo interesse, apoio e decisiva participação;

Aos Engenheiros João José Neto, Ting Kong Sen e Selma Shin Shimizu Melnikoff pelo incentivo e valiosas sugestões apresentadas;

Ao Prof. Dr. Antonio Hêlio Guerra Vieira e à "Fundação para o Desenvolvimento Tecnológico da Engenharia" que me colocaram à disposição todos os recursos necessários para a confecção deste trabalho;

À Sra. Judite Peres de Souza, minha esposa, pela dedicação e colaboração na preparação dos textos;

À Sra. Sonia Regina Izarelli pelos trabalhos de datilografia;

Ao Humberto Neservilha pelos serviços de impressão.

A todos os engenheiros e estagiários da F.D.T.E. e do Laboratório de Sistemas Digitais que direta ou indiretamente contribuíram para a preparação e testes dos programas relacionados com este trabalho.

## R E S U M O

Este trabalho descreve a implementação de um Núcleo de programação de um Sistema Básico de Controle para o Minicomputador "PATINHO FEIO" desenvolvido no "Laboratório de Sistemas Digitais do Departamento de Eletricidade da Escola Politécnica da Universidade de São Paulo".

São consideradas inicialmente as características mais importantes dos principais tipos de Sistemas Operacionais disponíveis em minicomputadores. São apresentadas algumas técnicas utilizadas na implementação de um Sistema Básico de Controle para o minicomputador G-10, desenvolvido na "Fundação para o Desenvolvimento Tecnológico da Engenharia".

Alguns conceitos relacionados com o desenvolvimento de Sistemas Operacionais em geral são comentados no capítulo 1.

O Capítulo 2 contém a descrição de um conjunto de recursos de programação denominado "NÚCLEO" proposto como ferramenta inicial para a implementação de sistemas operacionais no "PATINHO FEIO".

Em seguida é apresentada a descrição de um Sistema Básico de Controle baseado neste "NÚCLEO".

O apêndice 1 contém um resumo das principais subrotinas utilizadas na programação do "NÚCLEO".

No apêndice 2 é mostrado um exemplo de utilização de Sistema Básico de Controle.

## ABSTRACT

This work describes the implementation of a programming Nucleus and a Basic Control System for the minicomputer "PATIMBO FEIO" developed in the "Laboratório de Sistemas Digitais do Departamento de Eletricidade da Escola Politécnica da Universidade de São Paulo".

First, some important characteristics of the principal types of operating systems available for minicomputers are considered. Some techniques used in the implementation of a Basic Control System for the minicomputer G-10 developed in "Fundação para o Desenvolvimento Tecnológico da Engenharia" are presented.

Some general concepts related with the development of operating systems are commented in chapter 1.

Chapter 2 contains the description of a set of programming resources called "Nucleus". The Nucleus is proposed as a tool for the implementation of operating systems in the "PATIMBO FEIO".

Then a description of a Basic Control System based in this "Nucleus" is presented.

Appendix 1 contains a summary of the principal subroutines used in programming the "Nucleus".

In appendix 2 an example of the Basic Control System operation is shown.



# Í N D I C E

	PÁG.
1. INTRODUÇÃO	
1.1 - SISTEMAS OPERACIONAIS PARA MINICOMPUTADORES	3
1.1.1 - SISTEMA BÁSICO DE CONTROLE (SBC)	4
1.1.2 - SISTEMA OPERACIONAL COM DISCO	4
1.1.3 - SISTEMAS OPERACIONAIS DE TEMPO REAL	5
1.1.4 - SISTEMAS OPERACIONAIS DE TEMPO REAL C/DISCO	6
1.2 - IMPLEMENTAÇÃO DE UM SISTEMA BÁSICO DE CONTROLE	6
1.2.1 - CARACTERÍSTICAS ESSENCIAIS DO MINICOMPUTADOR G-10	7
1.2.2 - ORGANIZAÇÃO DO SISTEMA BÁSICO DE CONTROLE DO G-10	9
1.2.3 - TRATAMENTO DE INTERRUPÇÕES E MUDANÇA DE CONTEXTO	11
1.2.4 - FUNÇÕES DO SISTEMA BÁSICO DE CONTROLE DO G-10	14
1.2.5 - COMANDOS DO SISTEMA BÁSICO DE CONTROLE DO G-10	18
1.3 - ASPECTOS DO PROJETO DE UM SISTEMA OPERACIONAL	19
1.3.1 - PRINCIPAIS CARACTERÍSTICAS DO NÚCLEO	21
1.3.2 - O CONFLITO DE PROCESSO	23
1.3.3 - MULTIPROCESSOS	24
2. DEFINIÇÃO E IMPLEMENTAÇÃO DE UM NÚCLEO NO "PATINHO FEIO"	
2.1 - O ESQUEMA DE INTERUPÇÃO DO "PATINHO FEIO"	27
2.2 - PRIORIDADES E ROTINA DE DESCOBRIMENTO DE INTERRUPÇÕES	30
2.3 - INTERRUPÇÕES SÍNCRONAS E ASSÍNCRONAS	30
2.4 - PRIMITIVOS E PROCESSOS	33
2.5 - ESTADOS DE UM PROCESSO	35
2.6 - IMPLEMENTAÇÃO DO NÚCLEO	38
2.6.1 - ROTINA DE DESCOBRIMENTO DE INTERRUPÇÕES	40
2.6.2 - ROTINA DE TRATAMENTO DE INTERRUPÇÕES SÍNCRONAS	49



	PAG.
2.6.2.1 - GERAÇÃO DA INTERUPÇÃO SÍNCRONA.	41
2.6.2.2 - ESTRUTURA DE DADOS DA ROTINA ISS	42
2.6.2.3 - ESTRUTURA DA ROTINA ISS	42
2.6.3 - TRATAMENTO DAS INTERUPÇÕES ASSÍNCRONAS ..	49
2.6.4 - AÇIONAMENTO DE PROCESSOS	50
2.6.4.1 - PRIMITIVOS RELACIONADOS COM O RELÓGIO DE TEMPO REAL .....	53
2.6.5 - CANAL CONCENTRADOR DE ENTRADA E SAÍDA ....	54
2.6.5.1 - PRIMITIVOS DO CANAL CONCENTRADOR DE ENTRADA E SAÍDA .....	58
2.6.6 - COMUNICAÇÃO ENTRE PROCESSOS .....	63
2.6.6.1 - PRIMITIVOS PARA A COMUNICAÇÃO ENTRE PROCESSOS .....	65
2.6.6.2 - SINCRONIZAÇÃO DE PROCESSOS ....	67
2.6.7 - CONTROLE DE UM PROCESSO .....	68
2.6.7.1 - PRIMITIVOS DE CONTROLE .....	69

## 4. UM SISTEMA BÁSICO DE CONTROLE PARA O "PATINHO FEIO"

4.1 - ORGANIZAÇÃO DO SISTEMA BÁSICO DE CONTROLE .....	73
4.1.1 - ORGANIZAÇÃO DA MEMÓRIA PRINCIPAL .....	74
4.1.2 - INICIAÇÃO DO SISTEMA BÁSICO DE CONTROLE..	76
4.1.2.1 - O PROGRAMA CARREGADOR ABSOLUTO.	76
4.2 - DESCRIÇÃO DO PROCESSO SBC .....	78
4.2.1 - INICIAÇÃO DA INTERFACE 1 .....	78
4.2.2 - INICIAÇÃO DO RELÓGIO DE TEMPO REAL .....	78
4.2.3 - OBSERVAÇÕES SOBRE O PROCESSO SBC .....	80
4.3 - DESCRIÇÃO DO PROCESSO CONTROLADOR DE ENTRADA/ SAÍDA .....	80
4.3.1 - ESTRUTURA DOS DADOS DO PROCESSO PES .....	80
4.3.2 - PARÂMETROS DE COMUNICAÇÃO DO PROCESSO PES	82
4.3.3 - ESTRUTURA DO PROCESSO PES .....	84
4.4 - DESCRIÇÃO DO PROCESSO DE CARGA DE PROGRAMAS .....	86
4.4.1 - PARÂMETROS DE COMUNICAÇÃO DO PROCESSO CPB .....	87

	PAG.
3.4.2 - ESTRUTURA DOS DADOS DO PROCESSO CPE .....	90
3.4.3 - ESTRUTURA DO PROCESSO CPE E ALOCAÇÃO ....	90
3.5.4 - PRODUÇÃO DOS ARQUIVOS DE PROGRAMAS .....	93
3.5 - DESCRIÇÃO DO PROCESSO DE COMUNICAÇÃO COM .....	98
3.5.1 - DIRETIVAS EMPREGADAS NO PROCESSO COM .....	98
3.5.2 - ESTRUTURA DE DADOS DO PROCESSO COM .....	101
3.5.3 - ESTRUTURA DO PROCESSO COM .....	103
4. OBSERVAÇÕES FINAIS	
4.1 - MODIFICAÇÕES E AMPLIAÇÕES DO NÚCLEO BÁSICO .....	105
4.1.1 - PRIMITIVOS E PROCESSOS .....	106
4.1.2 - OPERAÇÕES DE ENTRADA E SAÍDA .....	107
4.1.3 - PROTEÇÃO .....	108
4.1.4 - EXPANSÃO DE MEMÓRIA .....	109
4.2 - AVALIAÇÃO DO SISTEMA BÁSICO DE CONTROLE .....	109
APÊNDICE 1 - DESCRIÇÃO DAS ROTINAS UTILIZADAS .....	111
APÊNDICE 2 - EXEMPLO .....	135

CAPITULO I - INTRODUÇÃO

## INTRODUÇÃO

O objetivo deste trabalho é descrever a implementação de um Sistema Básico de Controle e de um Núcleo de Programação destinados ao estudo e desenvolvimento de Sistemas Operacionais baseados no minicomputador "PATRIMIO FEIO".

O minicomputador "PATRIMIO FEIO" desenvolvido no LABORATÓRIO DE SISTEMAS DIGITAIS DO DEPARTAMENTO DE ELETRICIDADE DA ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO, descrito na referência (1), tem sido até então utilizado principalmente com o propósito de promover o treinamento de pessoal estagiário, tanto na área de "HARDWARE", através de modificações no projeto original ou inclusão de novos recursos, quanto na área de engenharia de "SOFTWARE" através da implementação de vários programas, notadamente na categoria de "SOFTWARE" de desenvolvimento(2).

Essas características justificaram a escolha do "PATRIMIO FEIO" como ferramenta na aplicação de vários conceitos atualmente utilizados na implementação de sistemas operacionais, a despeito de uma possível deficiência de alguns recursos de "HARDWARE" deste minicomputador. Entretanto, as soluções que possam ser dadas no sentido de eliminar essas deficiências foram consideradas como parte deste estudo, podendo proporcionar subsídios para outros projetos nesta área.

A definição dada ao NÚCLEO descrito no capítulo 2 procura abordar os principais aspectos envolvidos no estabelecimento dos recursos mínimos de programação que possibilitam o desenvolvimento de um sistema operacional, dentro das possibilidades de um minicomputador do porte do "PATRIMIO FEIO".

O Sistema Básico de Controle cuja descrição é dada no capítulo 3, tem principalmente duas finalidades, quais sejam, a de ilustrar a aplicação dos recursos proporcionados pelo Núcleo, no desenvolvimento de um sistema operacional simples como é o Sistema Básico de Controle, e ainda, por meio deste, avaliar o desempenho do Núcleo proposto inicialmente. Esta avaliação deverá



orientar as possíveis modificações, otimizações ou introdução de novos recursos, tanto de "SOFTWARE" como de "HARDWARE" que viabilizem a construção de sistemas operacionais mais sofisticados que levem em conta um amplo conjunto de aplicações.

## 1.1 - SISTEMAS OPERACIONAIS PARA MINICOMPUTADORES.

Sob um ponto de vista mais amplo, um sistema operacional pode ser considerado como sendo um conjunto de procedimentos manuais e automáticos que permitem a um grupo de pessoas compartilhar eficientemente os recursos de um computador (1). Se considerada essa definição, dentro a variedade dos aspectos envolvidos na análise e projeto de um sistema operacional, destacam-se os procedimentos automáticos destinados ao controle dos recursos de "HARDWARE" e "SOFTWARE" disponíveis no computador, atribuição desses recursos entre os vários programas usuários e ao estabelecimento de uma interface entre o "HARDWARE" e o "SOFTWARE". Estes procedimentos são normalmente atribuídos ao "SOFTWARE EXECUTIVO" (2). Isto é, ao conjunto de programas do Sistema Operacional encarregado desta tarefa.

Em se tratando de minicomputadores, o "SOFTWARE EXECUTIVO" assume uma importância considerável dentro do sistema operacional, podendo mesmo ser confundido com este. Neste caso, a natureza limitada dos recursos disponíveis e a variedade de aplicações dadas aos minicomputadores têm acarretado o surgimento de alguns tipos de "SOFTWARE EXECUTIVO", orientados de acordo com a natureza desses recursos e com a sua aplicação mais imediata.

De acordo com a referência (3), os sistemas operacionais para minicomputadores podem ser classificados em: Sistemas Operacionais "STAND-ALONE" ou Sistema Básico de Controle, Sistema Operacional com Disco (DISC-DISK OPERATING SYSTEM), Sistema Operacional de Tempo Real (RTOS - "REAL-TIME DISK OPERATING SYSTEM").

Algumas características mais importantes com respeito ao "SOFTWARE EXECUTIVO" associado a cada um destes sistemas serão apresentadas a seguir.

### 1.1.1 - SISTEMA BÁSICO DE CONTROLE (SBC)

Às vezes chamado de "Sistema Operacional de Fita de Papel" (1), um sistema "STAN-ALINE" dispõe de um conjunto de dispositivos de ENTRADA/SAÍDA do tipo teleimpressora (TTY), leitora e perfuradora de fita de papel e eventualmente impressoras de linhas e unidades de fita magnética. Estes dispositivos são utilizados por programas dedicados a tarefas simples, como por exemplo, edição de textos, conversão de dados, etc..

O controle destes dispositivos requer normalmente a assistência de programas especiais (Volantes ou "DRIVERS" de Entrada/Saída) cuja complexidade depende da existência ou não de processadores de Entrada/Saída capazes de automatizar por "HARDWARE" a maioria das operações envolvidas no controle destes dispositivos.

Além dessa assistência, a facilidade de programação das operações, mesmo com a inclusão de novos dispositivos, constitui uma das principais funções do "SOFTWARE EXECUTIVO".

O Sistema Básico de Controle pode servir de ponto de partida no desenvolvimento de outros sistemas operacionais se incorporar, além das funções de Entrada/Saída, outros procedimentos básicos necessários no estabelecimento de interface entre o "HARDWARE" e o "SOFTWARE".

### 1.1.2 - SISTEMA OPERACIONAL COM DISCO

A inclusão de discos magnéticos com a capacidade de armazenar grandes quantidades de dados, acessíveis de modo aleatório, proporciona um novo tipo de recurso ao sistema que pode ser controlado pelo "SOFTWARE EXECUTIVO". A facilidade com que os programas usuários podem manipular esses dados é representada pelo sistema de arquivos (FILE SYSTEM) proporcionado nos Sistemas Operacionais com Disco (1).



A existência de um sistema de arquivos em disco proporciona o recurso da "Segmentação de Programas", isto é, a possibilidade de se executar programas que ocupam no total, uma área de armazenamento maior que a capacidade da memória principal. Com este recurso, através da seqüenciação dos vários segmentos que constituem um programa, o usuário pode utilizar comodamente os programas de desenvolvimento, em geral constituídos por vários passos, tais como, MONITADOR (Assembler), COMPILADORES, EDITORES, RELOCADOR-LIGADOR, etc, na confecção de seus próprios programas de aplicação. A seqüenciação dos segmentos, isto é, a troca entre segmentos residentes na memória principal por outros segmentos de arquivo de programas no disco (Swapping Systems) (3) constitui-se numa das principais atividades do "SOFTWARE EXECUTIVO" em um Sistema Operacional com disco.

Neste tipo de sistema, a atribuição dos recursos é realizada sequencialmente entre os diversos usuários, de tal forma que a totalidade dos recursos disponíveis é atribuída ao único usuário em cada instante, dependendo apenas de sua requisição através de diretivas enviadas ao "EXECUTIVO" ou "MONITOR".

#### 1.1.3 - SISTEMAS OPERACIONAIS DE TEMPO REAL

As aplicações que envolvem outros tipos de dispositivos de ENTRADA/SÁIDA, como por exemplo, conversores analógico-digitaís e digitais analógicos em um controle de processos ou aquisição de dados (6), a velocidade de resposta a estímulos externos (interrupções) provocadas por esses dispositivos, tornam-se um parâmetro importante no projeto do "SOFTWARE EXECUTIVO".

Normalmente, esta "resposta" consiste na execução de um determinado programa, escolhido de acordo com o estímulo; entre os vários programas residentes na memória principal.

Dada a natureza dos fenômenos destes estímulos, o "EXECUTIVO" deve tomar as providências necessárias na ocorrência de con-



lhos, por exemplo, estímulos simultâneos que exijam a execução de diferentes programas naquele instante. Se a cada programa se tiver associada uma prioridade, o "EXECUTIVO" deve resolver os seus conflitos baseando-se na seguinte ordem de prioridade:

Um outro tipo de dispositivo, chamado RELOJO DE TEMPO REAL, permite ainda que programas possam ser executados periodicamente ou em instantes determinados. De um modo geral, a escolha do próximo programa a ser executado (Scheduling) e a comunicação entre um operador e os programas em questão, constituem as principais tarefas de um "SOFTWARE EXECUTIVO" em um sistema operacional de Tempo Real.

#### 1.1.6 - SISTEMAS OPERACIONAIS DE TEMPO REAL COM DISCO

As características de um Sistema Operacional de Tempo Real e de um Sistema Operacional com Disco, mencionadas anteriormente, podem estar associadas em um único sistema de maior capacidade, chamado Sistema Operacional de Tempo Real com Disco.

A não ser em uma aplicação, por exemplo, de aquisição de dados na qual o disco é utilizado para coletar esses dados, por um período relativamente longo, as tarefas em "Tempo Real" são independentes das outras relacionadas com a execução de um programa.

Essa característica de "MULTIPROGRAMAÇÃO" introduz outros requisitos no "SOFTWARE EXECUTIVO", quais sejam, as de controlar a não interferência entre os programas executados em "Tempo Real" (FOREGROUND) e aqueles de menor prioridade (BACKGROUND) concorrendo com os recursos disponíveis no sistema.

#### 1.2 - IMPLEMENTAÇÃO DE UM SISTEMA BÁSICO DE CONTROLE (S.B.C.)

Os principais aspectos da implementação de um sistema operacional simples do tipo "REAL-TIME" desenvolvidos no mini-computador (S-10-112) são abordados neste item, com a finalidade

de ilustrar algumas técnicas que serão utilizadas na implementação de um sistema do mesmo tipo para o minicomputador "PATINHO FEIO".

Este sistema denominado SISTEMA BÁSICO DE CONTROLE foi projetado com as seguintes finalidades:

- a) uniformizar e facilitar a programação das operações de ENTRADA/SAÍDA;
- b) tratar as interrupções especiais;
- c) facilitar o teste de um conjunto de programas ligados ao "SOFTWARE BÁSICO" desenvolvido para o G-10, tais como, o MONTADOR (16), CARREGADOR-LIGADOR (16) e um Compilador FORTRAN, utilizando o próprio G-10.

Este sistema foi desenvolvido adotando-se uma configuração que inclui uma memória com 16k palavras de 16 bits, leitora e perfuradora de fita de papel, uma teleimpressora (TTY), leitora de cartões e uma impressora de linha.

#### 1.2.1 - CARACTERÍSTICAS ESSENCIAIS DO MINICOMPUTADOR G-10

As principais características do minicomputador G-10 (vercite na referência (12)), e que são essenciais no desenvolvimento de um Sistema Básico de Controle com os propósitos citados, podem ser resumidas da seguinte forma:

##### a) ENDEREÇAMENTO DA MEMÓRIA PRINCIPAL

A principal característica em relação ao endereçamento é a existência de endereçamento relativo com respeito a duas bases, a base de programas e a base de dados. Deste fato decorre a divisão de um programa em duas áreas lógicas, a área de códigos e a área de dados, referenciados por meio de estruturas distintas,

toda uma dessas áreas é limitada por um registrador LIMITE DO PROGRAMA e LIMITE DE DADOS respectivamente.

#### b) MODOS DE OPERAÇÃO

Dos modos de operação da Unidade Central de Processamento, chamados MODO SUPERVISOR e MODO USUÁRIO permitem que um conjunto de instruções especiais, constituídas por instruções do tipo, mudanças de bases, instruções de ENTRADA/SAÍDA, etc., seja executada somente no MODO SUPERVISOR. Em particular, neste modo, a base de programas é feita automaticamente igual a zero enquanto que a base de dados e limites podem admitir quaisquer valores.

#### c) OPERAÇÕES DE ENTRADA/SAÍDA

Os dispositivos de ENTRADA/SAÍDA conectados ao CANAL CONCENTRADOR, são controlados diretamente pela U.C.P., através de instruções de ENTRADA/SAÍDA executadas na transferência de cada dado destes dispositivos. Exceto a impressora de linha, os demais dispositivos utilizados no S.H.C., estão conectados ao CANAL CONCENTRADOR.

A impressora de linha está conectada ao CANAL SELETOR, a qual tem a capacidade de realizar a transferência de um bloco de dados com acesso direto à memória principal, interagindo com a U.C.P. somente no início e final da transferência do bloco de dados.

#### d) INTERRUPÇÕES

Qualquer interrupção da U.C.P. é caracterizada por uma mudança para MODO SUPERVISOR, armazenamento do CONTADOR DE INSTRUÇÕES (registrador RDI, BASE DE PROGRAMAS e PALAVRAS DE "START" nas três posições de memória consecutivas a partir daquela apontada pelo registrador REREGISTRADOR TOPO DA PILHA). Após estas operações, o CONTADOR DE INSTRUÇÕES é carregado com o valor contido em uma posição fixa da memória de acordo com a origem da interrupção.



As interrupções podem ser provocadas pelos canais de ENTRADA/SÁIDA, pela execução da instrução CSOP (denominada de "SUPERVISOR") ou por eventos especiais (14), como violação de memória, execução de instrução privilegiada em MODO USUÁRIO, falta de alimentação ou por meio do parrel.

#### 1.2.2 - ORGANIZAÇÃO DO SISTEMA BÁSICO DE CONTROLE DO L-10

O sistema Básico de Controle é constituído por um conjunto de rotinas executadas em MODO SUPERVISOR e suporta a existência de apenas um programa executado em MODO USUÁRIO.

A distribuição dos códigos e dados do S.B.C. e do programa usuário na memória principal são feitos de acordo com o esquema da fig. 1.1.

A área POSIÇÕES DE INTERUPÇÃO (0 a 80) é reservada para conter os endereços das rotinas de tratamento das interrupções, conforme o esquema de interrupção adotado.

O S.B.C. opera com base de programa (R.P.S.) igual a zero por ser executado em MODO SUPERVISOR. A utilização da base de dados (B.D.S.) também igual a zero permite que o S.B.C. tenha o acesso a toda memória como se esta fosse uma única área de dados.

Os dados propriamente ditos do S.B.C. foram reunidos nas primeiras posições que ocupam as POSIÇÕES DE INTERUPÇÃO (ÁREA DE DADOS DO S.B.C.) enquanto que os códigos estão agrupados na ÁREA DE CÓDIGOS DO S.B.C.

Os dados referenciados através do registrador RI (PURA) manipulados pelo S.B.C., por ser caráter dinâmico, isto é, por não ser previsível exatamente a quantidade de posições que poderão ocupar, foi colocada nas últimas posições de memória (31 000), sendo indicadas pela variável R15 (PI do Supervisor).

A possibilidade do S.B.C. ter acesso a totalidade da memória faz com que o registrador LIMITE DE DADOS assumam a função

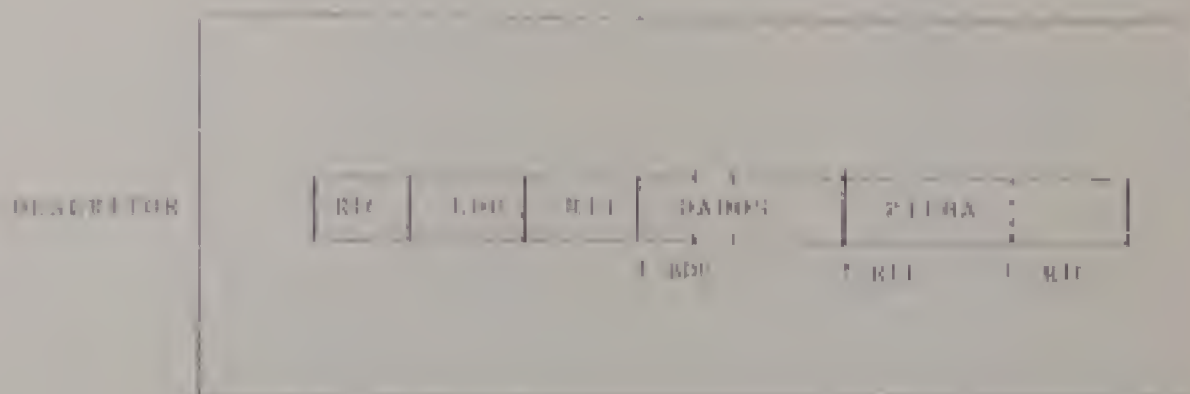


ORGANIZAÇÃO DE MEMÓRIA DO S.B.C. DO C-10

BPS, BDS	POSICÕES DE INTERUPÇÃO
	ÁREA DE DADOS DO S.B.C
	ÁREA DE CÓDIGOS DO S.B.C
BPU	ÁREA DE CÓDIGOS DO USUÁRIO
LPU	DESCRITOR
BDU	ÁREA DE DADOS DO USUÁRIO
R11	PILHA DO USUÁRIO
L11	ÁREA LIVRE
R15	PILHA DO S.B.C
L15	

que da última posição de memória disponível (1100) quando os registros do S.C.P. estiverem em funcionamento.

O programa controlador é constituído por ÁREA DE CÓDIGOS DO PROGRAMA (delimitada por uma base de programa (BPB) e um limite de programa (LPP)), ÁREA DE DADOS DO PROGRAMA, por sua vez, é constituída por um PREFIXO, DADOS E PALAVRA conforme ilustrado abaixo:



O PREFIXO é constituído por três palavras, contendo os valores do APONTADOR DA PALAVRA (RIP) CORRENTE (RIP), O LIMITE DE DADOS e o valor inicial do APONTADOR DA PALAVRA (RIP).

Este PREFIXO será utilizado na mudança de contexto.

#### 1.2.1 - TRATAMENTO DE INTERRUPÇÕES E MUDANÇA DE CONTEXTO

O CONTEXTO pode ser caracterizado pelos valores de: APONTADOR DA P.C.P., BASE DE PROGRAMAS, LIMITE DE PROGRAMAS, BASE DE DADOS, LIMITE DE DADOS, APONTADOR DA PALAVRA (RIP), PALAVRA DE "STATUS". As PALAVRAS DE "STATUS" em particular, estão correlacionadas de modo (Supervisor ou Privilegiado de Interrupção (1) da P.C.P., Cláusula em Inibição).

Dados contextos que utilizamos no S.H.C., o contexto do SUPERVISOR e o contexto USUÁRIO, nos quais os valores dos registradores são tabelados abaixo:

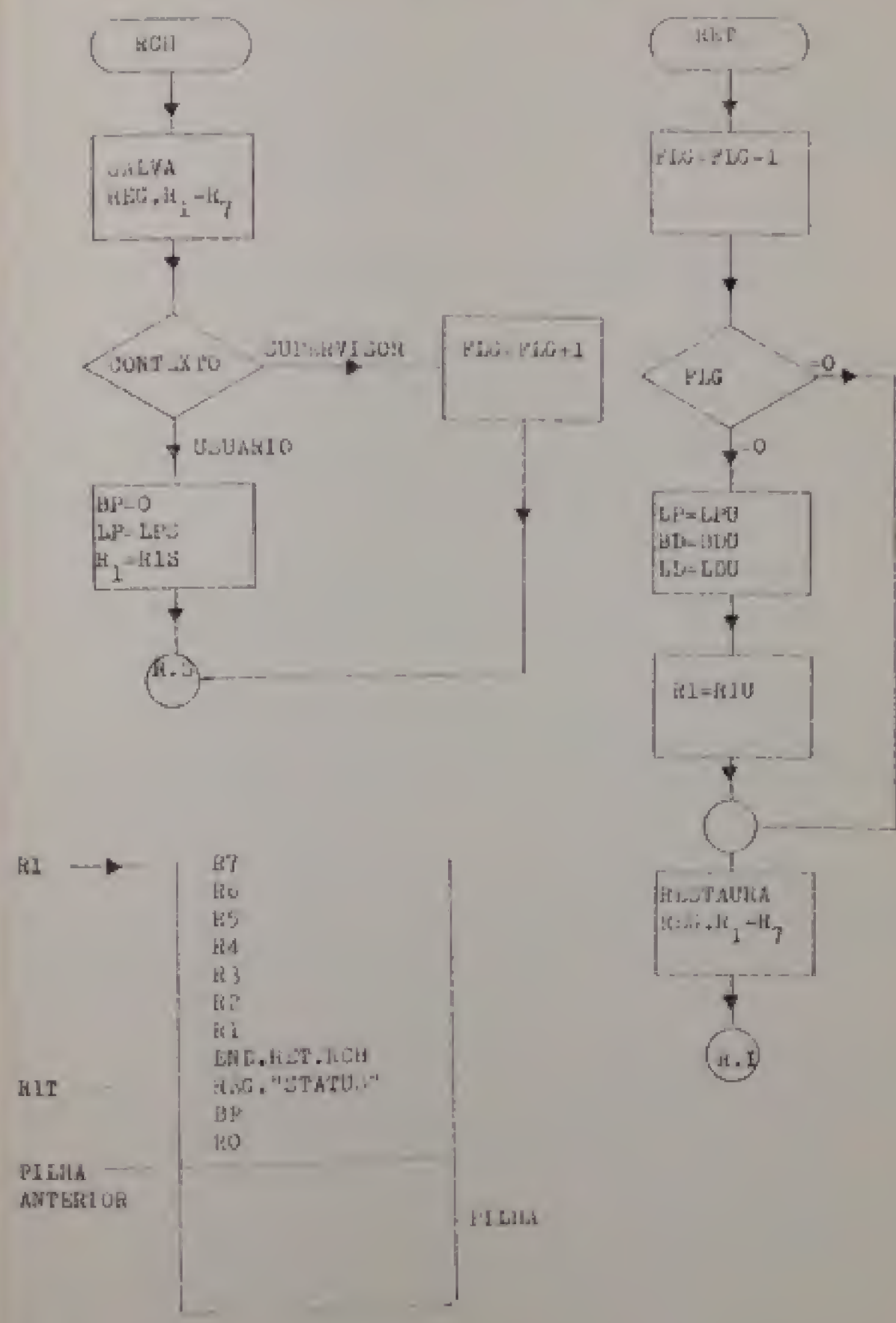
CONTEXTO SUPERVISOR		CONTEXTO USUÁRIO
RP	0	RPO
LP	LPN	LPO
BB	0	BPO
ED	(EDF) + 16	EDU
RT	RIN	RTU
"STATUS"	MODOS	MODOS, 1=0

Uma mudança de contexto ocorre por ocasião de uma interrupção ou no final do tratamento de uma interrupção. Essa mudança é em parte realizada automaticamente pelo "hardware" e em parte por duas rotinas específicas do S.H.C., chamadas RCH (rotina de mudança de contexto) e RFL (Rotina de interrupção).

A rotina RCH, utilizando uma variável de controle PLO, realiza a mudança para CONTEXTO SUPERVISOR se o contexto anterior à uma interrupção for USUÁRIO, de acordo com o diagrama da Figura 1.2.

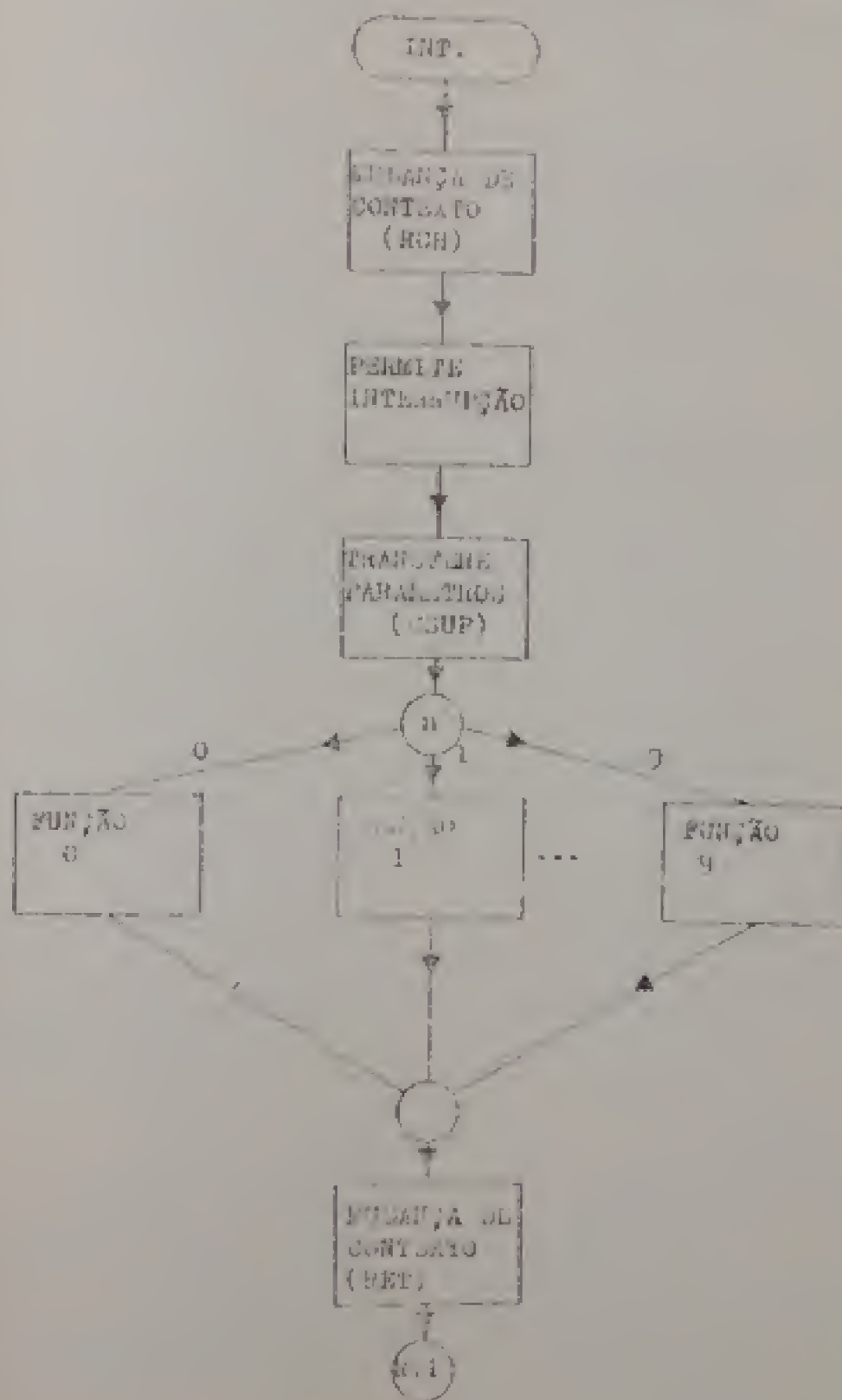
O tratamento de qualquer interrupção inicia-se por uma chamada de rotina RCH. Nesta ocasião, o CHRA (Supervisor ou Usuário) para a interrupção indicada na Figura 1.2, sendo RP, BB e "STATUS" armazenados pela interrupção, o endereço de retorno de RCH (LPO, RTT, RPO) pela instrução de chamada de subrotina (PRG R1, 04 PRG) e os valores de R1 a R7 armazenados pela instrução de SALVA REGISTRADORES (SVR R1).

Pelo exame de "STATUS" na PLO, obtém-se o MODOS do detector, o que vai determinar a mudança de contexto ou não. No caso de uma interrupção ocorrer no contexto SUPERVISOR, esta se resume "continua" na variável PLO, fazendo mudança de contexto DESCRITOR do USUÁRIO e a rotina segue com o contexto.













#### b) FUNÇÕES DE CONTROLE DO PROGRAMA CORRIGIDO

Para o controle do programa corrigido, as FUNÇÕES 02, 03 e 04, realizam o controle do programa anterior (CSPP 01 ou CSPP 03) ou a suspensão temporária de execução de qualquer comando do programa (CSPP 03) quando pela tecla impressora (COMP 03).

#### c) FUNÇÕES DE CONTROLE DO RASTREAMENTO

Uma característica do microcomputador G-10 é a existência de um tipo de interrupção, chamado RASTREAMENTO ("TRAC"), que pode ocorrer antes do início da execução de cada instrução pela R.C.P.

O rastreamento desta interrupção, no S.B.C., é realizado por meio de interrupção em um dispositivo de E/S, onde pode ocorrer, de conteúdos das variáveis de controle do S.B.C. Assim, quando um programa pode ser interrompido durante sua execução, permitindo o seu teste.

O controle das funções do programa nos quatro estados de funcionamento deve ser realizado, controlado por meio das funções de "LIGA RASTREAMENTO" (COMP 01), "DESLIGA RASTREAMENTO" (CSPP 02).

#### d) INTERPRETAÇÃO DE COMANDOS

A função 05 permite que o S.B.C. execute comandos em um modo resultante da interpretação de um comando digitado pelo usuário (COMP 05).

Assim, a interpretação

DESEMPENHO

passa a interpretação do comando codificado na forma de dados e é direcionada pelo código COM.

No conjunto de variáveis de controle, a função 06 permite que o S.B.C. execute comandos em um modo

## 1, 2, 3 - COMANDOS DO SISTEMA BÁSICO DE CONTROLE

A interação entre o Sistema Básico de Controle e o operador é estabelecida por meio de COMANDOS e MENSAGENS transmitidas por meio de uma teleterminal (TTY).

Esta interação caracteriza o S.B.C., cujo diagrama resumido está esquematizado na Fig. 1.3.

Esse conjunto de comandos, reunidos na tabela abaixo, representam as principais operações necessárias na execução de teste e execução de um programa.

As MENSAGENS, pelo "COMANDO NÃO RECONHECIDO" através do sinal (Y) e "COMANDO REALIZADO" (Z) permitem que esta interação se complete.

TABELA DE COMANDOS (24)

COMANDO	SIGNIFICADO
A	Termina programa em execução.
B	Lista bases e limites do programa.
C, N, X	Carrega programa da unidade B e reserva X posições para a pilha.
EP, E, Lista	Modifica as posições da área de programas a partir do endereço E com os valores da Lista.
IP, Lista	Acessa os valores constantes da Lista na área de programas.
IP, Lista	Acessa os valores constantes da Lista na área de dados.
EP, N, X, Y	Lista na unidade B a área de programas e reserva endereços X e Y.

$ED, R, X, Y$	Lista de endereços que contém os dados, entre os endereços $X$ e $Y$ .
$P, E$	Compara o endereço $P$ com o endereço $E$ e produz o resultado de comparação de endereços $P$ e $E$ .
$R$	Endereço do programa a ser executado.
$E, R, X, Y$	Endereço $E$ para o endereço $R$ na unidade lógica $X$ e $Y$ na unidade de endereços $X$ e $Y$ .
$P, X, Y$	Toma unidades lógicas $X$ e $Y$ para $P$ .
$V$	Continua execução do programa.

### 1.3 - ASPECTOS DO PROJETO DE UM SISTEMA OPERACIONAL

A seguir os aspectos de projeto de um sistema operacional são apresentados sob o ponto de vista de uma grande variedade de aspectos, desde um sistema operacional simples, como um SISTEMA BÁSICO DE CONTROLE até um Sistema Multitarefa, passando como o de um SISTEMA DE TEMPO-REAL, COM SÍNCRONIA.

Entretanto, o principal problema no projeto de um sistema operacional é a escolha dos elementos de projeto, que, muitas vezes, são dados e que não podem ser mudados, de forma que não tenham liberdade de programação que permita uma expansão de acordo com as necessidades do sistema (12).

Principalmente quando se trata de microcomputadores, a capacidade de expansão futura é muito importante, uma vez que pode ocorrer um paralelo com a evolução de nossos computadores de "Hardware", como, por exemplo, de um disco magnético, sendo, como um computador, como pode ser notado na evolução do hardware. Tem-se apenas a limitação de microcomputadores, a limitação dos dados disponíveis, uma grande expansão por expansão de hardware.

Entretanto, a capacidade de expansão é limitada, com que um dado sistema operacional possa ser desenvolvido, possibilitando a evolução de um sistema, a partir de uma configuração de hardware.



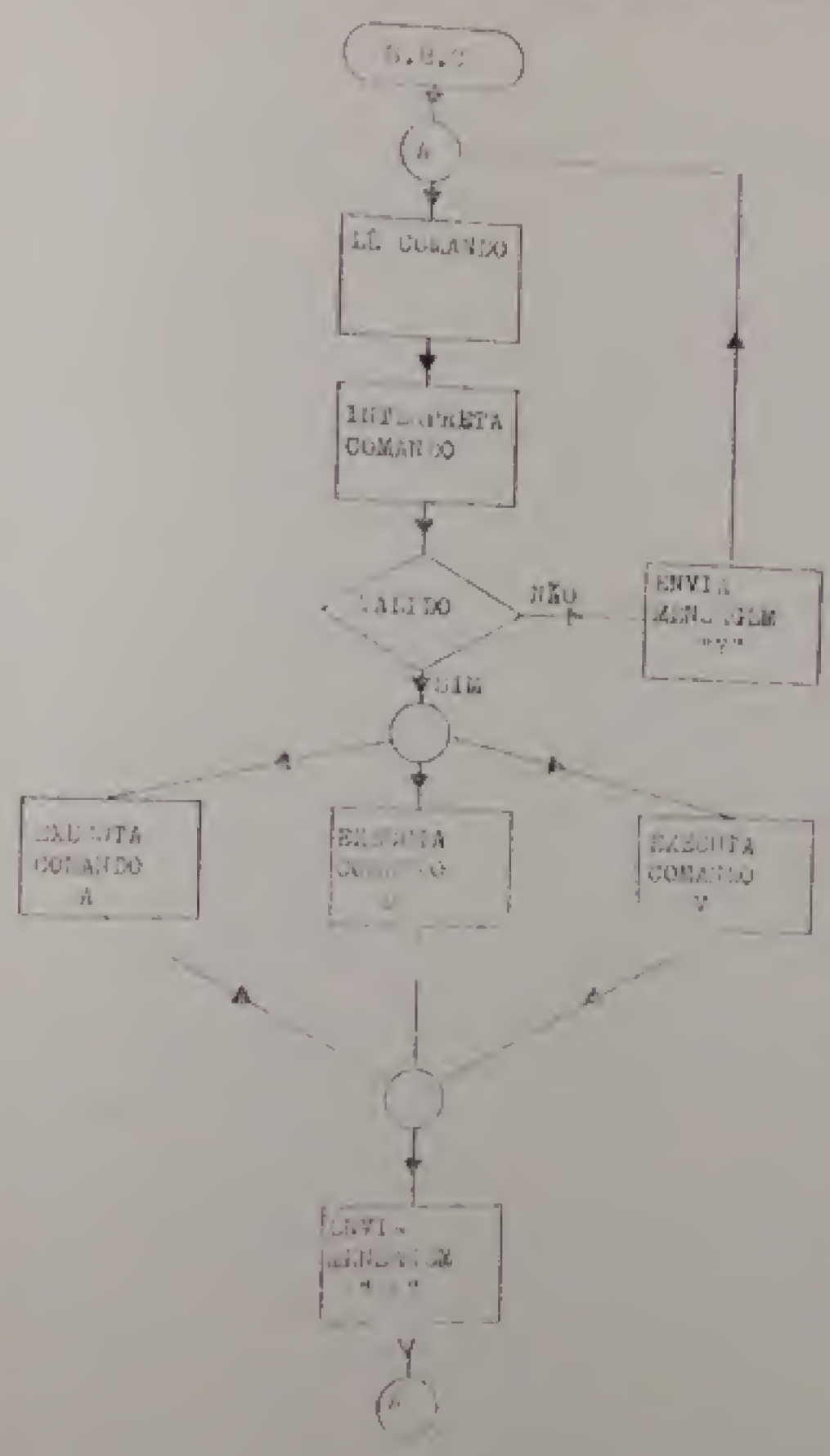


Fig. 1.1

conduzem à necessidade de se ter um implemento, ou com carácter rígido ou modulável.

Uma forma de se organizar os recursos aqueles que controlam o sistema operacional pode ser vista na estrutura (10) a qual será adotada neste trabalho.

Basicamente, como ilustra a figura 1.5, o primeiro módulo é constituído por um NÚCLEO, o qual concentra os recursos básicos de "Hardware" tais como a UNIDADE CENTRAL do PRIMEIRO NÍVEL, I/O, P.P., processadores de ENTRADA/SÁIDA, conjunto de instruções de máquina, memória principal, etc., além de "Software" necessários ao estabelecimento da interface entre o "Hardware" e o restante do sistema.

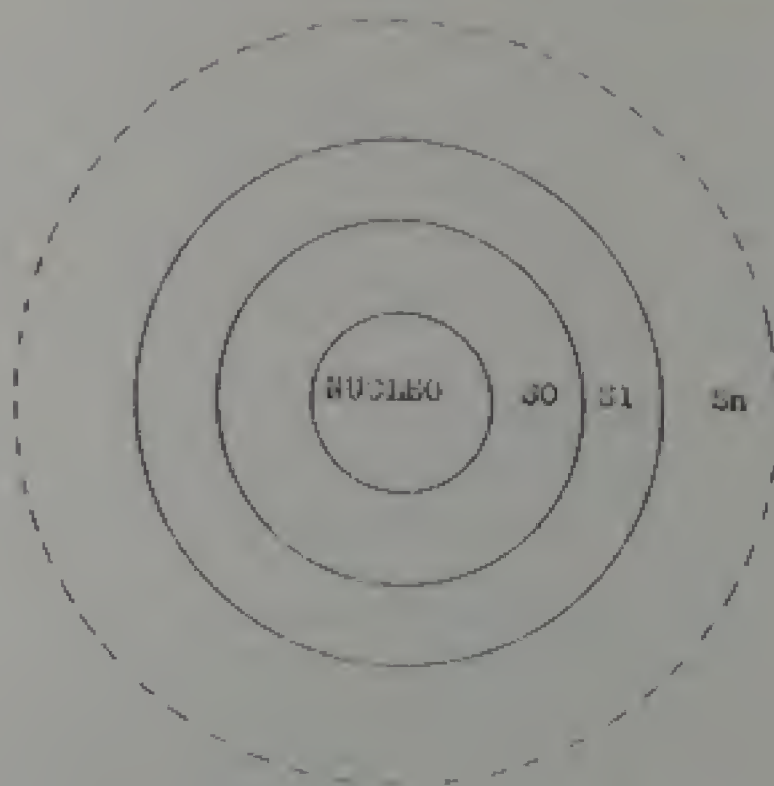
No segundo nível estão os processos que utilizam apenas os recursos do Núcleo implementando um sistema  $S_n$ .

A expansão se realiza representando novos níveis, em que se utilizam os recursos do primeiro nível mais interno. Assim, a implementação de um sistema operacional  $S_n$  é realizada utilizando-se todos os recursos de um sistema  $S_{n-1}$ .

#### 1.4.1 - PRINCIPAIS CARACTERÍSTICAS DO NÚCLEO

O NÚCLEO de um sistema operacional pode ser caracterizado como sendo um conjunto de dispositivos capazes de executar uma série de funções, em uma sequência bem determinada.

Fazendo parte do NÚCLEO, o I/O, P.P., possui as seguintes funções de aquisição importante: possivelmente um processador de ENTRADA/SÁIDA, executando várias instruções no controle de um ou mais dispositivos periféricos. No nível interno cada computador é caracterizado por uma série de recursos que dependem do sistema definido. Estes recursos podem ser classificados por exemplo a seguir de acordo de duas modalidades: DE NÍVEL DE DADOS ou DE NÍVEL DE UM NÍVEL DE COMANDO, por um dispositivo de ENTRADA/SÁIDA.





Na execução do que programado, um certo número de instruções de alguns de comandos pela ocorrência de interrupções de máquina que compõem tal programa, e que podem ser devidas a qualquer momento da execução, devido a possibilidades de ocorrência de INTERRUPTIOES (E.I.), e que interrupções são normalmente causadas por eventos regulares, com a execução das instruções e são devidos, por exemplo, a terminos de operação em dispositivos de ENTRADA/SÁIDA ou condições anormais resultantes da tentativa de programar utilizar recursos não autorizados.

O atendimento desses interrupções é normalmente realizado por outro sequencia de instruções não diretamente relacionada com o programa em andamento.

Na maioria dos microcomputadores, são previstas instruções em situações especiais que permitem a execução de uma série de instruções de um modo não interrompível. Este fato pode ser utilizado na implementação de subprogramas não interrompíveis chamados PRIMEIROS.

Um PRIMEIRO se processa como se fosse uma instrução de máquina, possibilitando por isso, ao CPU, uma ampliação da capacidade de processamento de 4096 através do emprego de um conjunto adequado de PRIMEIROS.

Do ponto de vista de hardware, no tipo 80, o 80C100 apresenta portanto como uma "80C100 VIRTUAL" com capacidade superior à do "80C100" que seria realmente.

### 1.1.2 O CONHECIMENTO "SEGURADO"

Um programa pode ser caracterizado como sendo a descrição por meio de códigos da linguagem de máquina, de uma série de ações a serem realizadas pelo CPU. Uma vez localizado em alguma área da memória principal, para que este programa possa ser executado, é necessário que o CPU, assumindo determinado "NÍVEL" que pode ser definido pelos valores dos registradores de

propósito geral, CONTEÚDO DE INSTRUÇÕES, contendo de 1 a 1000. O registrador de "STATUS" do FLUXO de DADOS, Modo de Operação do B.C.P., etc., dependendo da particular C.C.P., é dada a que está anexa as instruções indicadas pelos códigos do programa, e os acionamentos "STATUS" que são ligados ao "STATUS" anterior e de cada instrução subsequente.

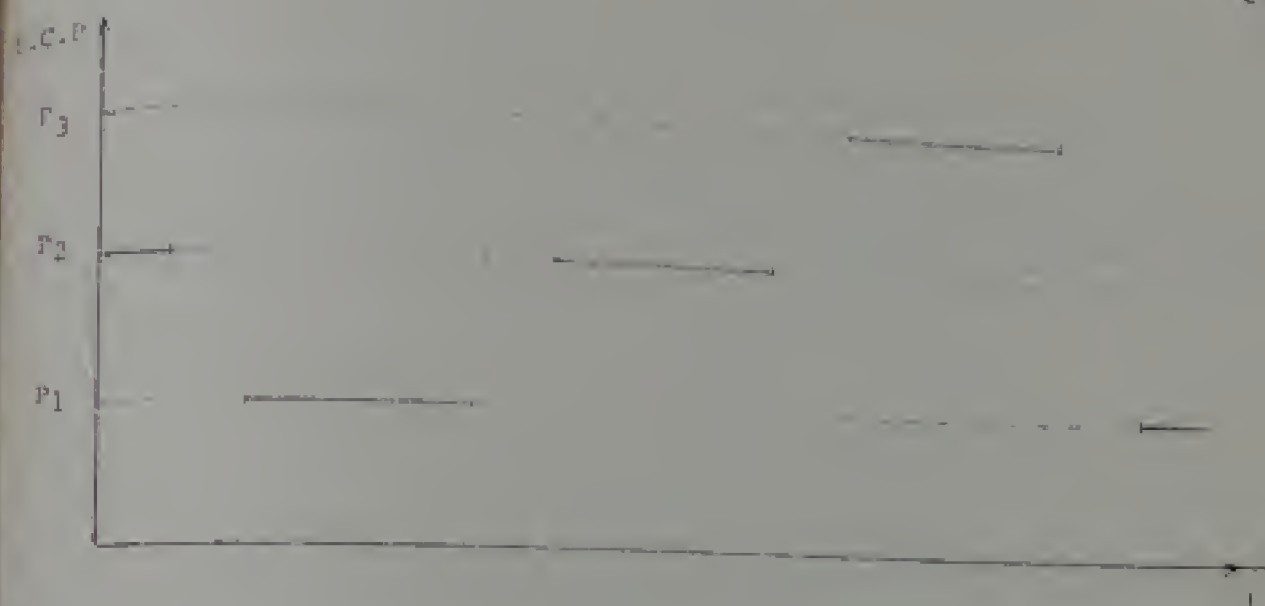
Uma vez que tal sequência de instruções pode ser interrompida, alguma providência deve ser tomada no sentido de salvar esse "STATUS" quando ocorrer a interrupção, para que o programa em execução possa ser retomado corretamente após o atendimento da interrupção. Associando-se o valor do "STATUS" do B.C.P. de cada instante ao programa em execução, poderá conceber-se um PROCESSO como sendo uma sequência de estados  $A_0, A_1, \dots, A_i, A_{i+1}, \dots$  e uma sequência de estados  $S_0, S_1, \dots, S_i, S_{i+1}, \dots$  de um programa em execução tal forma que um estado  $S_{i+1}$  é determinado por  $A_{i+1}$ , e o estado  $A_{i+1}$  é função do estado  $S_i$  (1).

Uma vez que um processo não constituir-se no elemento fundamental na implementação de um sistema operacional, é necessário que em cada caso seja dada uma definição precisa e objetiva para este conceito.

#### 1.3.3 - MULTITASKING

Uma vez que um processo pode ser interrompido e retomado posteriormente, a técnica de MULTIPROCESSING (MULTITASKING) é frequentemente utilizada na implementação de um Sistema Operacional. Por isto, o sistema admite a existência de vários processos simultâneos utilizando os recursos do sistema, daí o termo "MULTIPROCESSOS".

De qualquer modo, a B.C.P. é cedida a cada um dos processos e estes a utilizam por um determinado período de tempo, até que algum evento ocorra que interrompa este processo. O diagrama abaixo ilustra a utilização da B.C.P. de acordo com uma utilização por três processos  $P_1, P_2$  e  $P_3$ .



Pode-se observar que a computação da U.C.P. entre um processo e outro não ocorre instantaneamente, exigindo um certo intervalo de tempo ("OVERHEAD") no salvamento do "STATUS" associado a  $P_1$  e a recuperação do "STATUS" de  $P_2$ , por exemplo, no qual a U.C.P. é utilizada para isto.

A técnica de MULTIPROCESSOS visa otimizar o tempo de uso da U.C.P., uma vez que, naturalmente, um processo pode ficar bloqueado aguardando, por exemplo, a execução de uma operação de ENTRADA/SAÍDA (I/O). Enquanto isto, pode existir algum outro processo independente em condições de utilizar a U.C.P.

Em outros casos, a utilização de MULTIPROCESSOS decorre naturalmente do tipo de aplicação do Sistema Operacional, como é o caso de um sistema de Tempo-Real.

Para que um Sistema Operacional realize as funções para as quais está destinado, utilizando a técnica de MULTIPROCESSOS, será necessário que este possua a capacidade de sincronização e comunicação entre os vários processos e a possibilidade de adição e remoção de processos de um modo dinâmico (3).



Quando se tem interferência de autorproteção, como nos *Sistemas de Tempo e de Espaço*, a existência de proteção concorre para a conservação da diversidade no sistema, mesmo para a necessidade da inclusão de processos relacionados com a atribuição e controle de recursos de uma forma que possa garantir a não interferência entre as atividades independentes do mesmo *Sistema Operacional*.

CAPÍTULO 2 : DEFINIÇÃO E IMPLEMENTAÇÃO DE  
UM NOVO NO "PATRÃO FILO"

## 2. DEFINIÇÃO E IMPLEMENTAÇÃO DE UM NÍVEL NO "PATINHO FEIO"

Conceitualmente, o NÍVEL será constituído por um conjunto de programas e dispositivos do "PATINHO FEIO" capazes de implementar um determinado conjunto de ações denominadas PRIMITIVOS. Estes primitivos incluirão os conjuntos de instruções de máquina do "Patinho Feio" e outros que serão relacionados neste capítulo. Geralmente, com a finalidade de caracterizar objetivamente o conceito de primitivo, alguns detalhes do "Hardware" do "Patinho Feio" serão evidenciados.

### 2.1 - O ESQUEMA DE INTERRUÇÃO DO "PATINHO FEIO"

Os detalhes do esquema de interrupção no "Patinho Feio" estão definidos nas setecentas (11 a 17). Um modelo simplificado deste esquema será adotado aqui para efeito de referência, tratando apenas seus aspectos funcionais, (Fig. 2.1).

Existe a possibilidade de se conectar até 16 interfaces à M.C.P., em posições numeradas de 1 a 16. Em cada interface  $i$ , existe um conjunto de sinais que participam da lógica de interrupção dos quais se destacam os sinais ESTADO (EST), PERMITE INTERRUÇÃO DA INTERFACE (PRI $_i$ ) e PEDIDO DE INTERRUÇÃO DA INTERFACE (PDI $_i$ ).

Estes sinais estão relacionados por:  $(PRI)_i = (EST)_i \wedge (PRI)_i$ . Assim, a ocorrência de um pedido de interrupção à M.C.P., sem que o ESTADO atinja nível 1, não gera interrupção efetiva nem nível 1  $(PRI)_i \neq 1$ . Para os sinais podem ser alterados por instruções executadas na M.C.P.

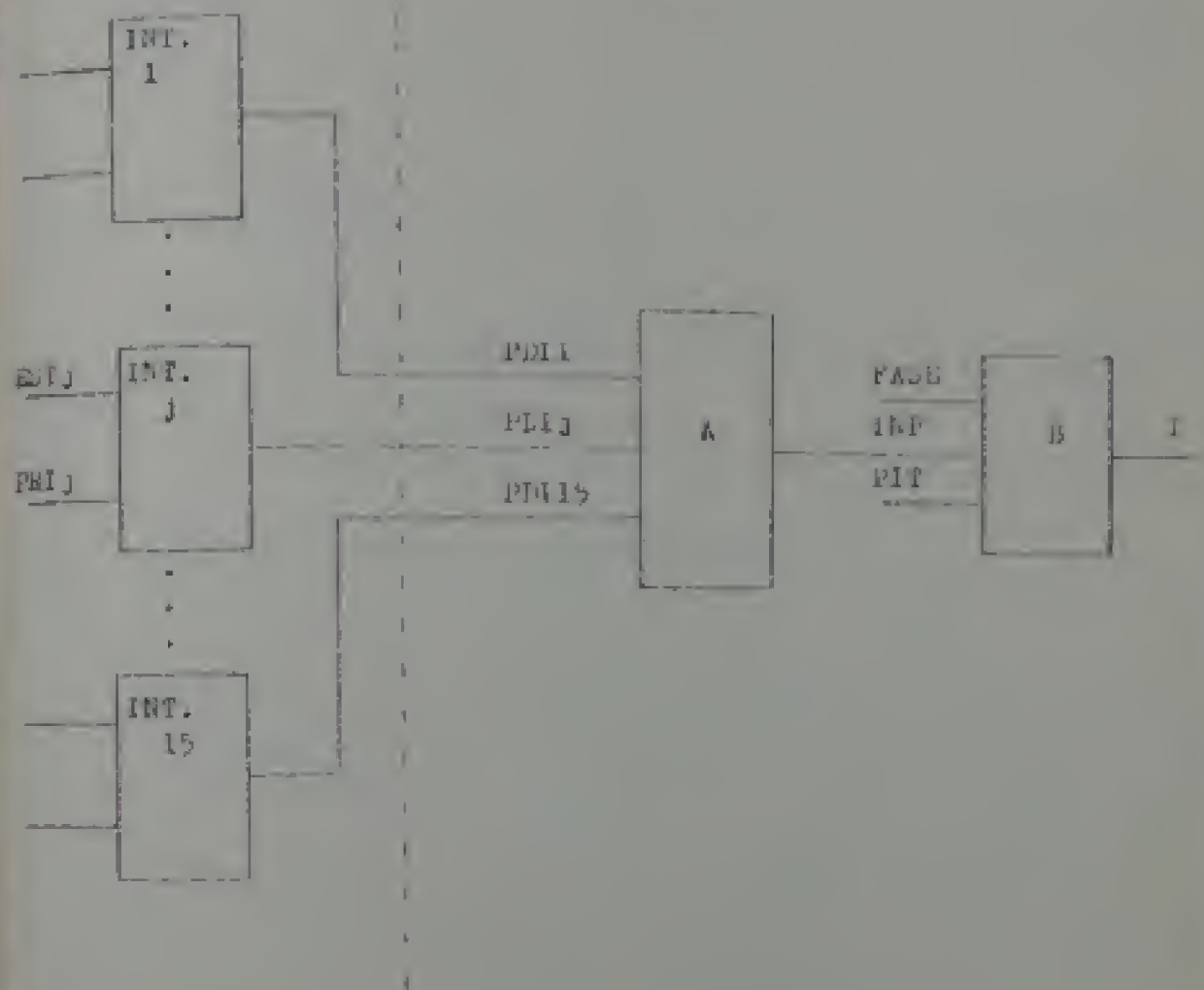
Os sinais de pedido de interrupção são inseridos à M.C.P. e representados no bloco lógico A que, através de uma lógica "AND" produz o sinal de INTERRUÇÃO PRESENTE (INP). O sinal INP não depende de escalonamento ou de prioridades, ou seja:

$INP = (PDI)_1 \vee \dots \vee (PDI)_i \vee \dots \vee (PDI)_n$ . Esse esquema não apresenta nenhuma prioridade com relação aos pedidos de interrupção de forma que no seu estado atual há apenas uma interrupção produzida.

# SYSTEM OF INTERFACED MODULES

INTERFACES

O.C.F.







A transição do estado INTERRUPTO para o estado NORMAL somente é realizada por meio do recebimento de uma instrução de estado denominada PDI. Esta instrução faz com que o CONTADOR DE INSTRUÇÕES recorra ao vetor contido nas posições de endereços 2 e 3 da memória, proporcionando o início da execução da instrução indicada por esse endereço e ainda,  $INT = 0$ , realizando a transição para o estado NORMAL.

## 2.2 - PRIORIDADES E ROTINA DE DESENVOLVIMENTO DE INTERUPÇÕES

Quando ocorre uma interrupção, a primeira providência a tomar é descobrir qual a interface que provocou tal interrupção. Isto é realizado por um componente do MÓDULO denominado ROTINA DE DESENVOLVIMENTO DE INTERUPÇÕES, uma vez que essa informação não é imediatamente fornecida pelo "hardware".

A ROTINA DE DESENVOLVIMENTO DE INTERUPÇÕES (RDI) faz uso da instrução "TESTA SE HÁ PERÍODO DE INTERUPÇÃO DA INTERFACE" examinando um a um os sinais PDI, e diagrama da rotina pode ser visto na Fig. 2.2.

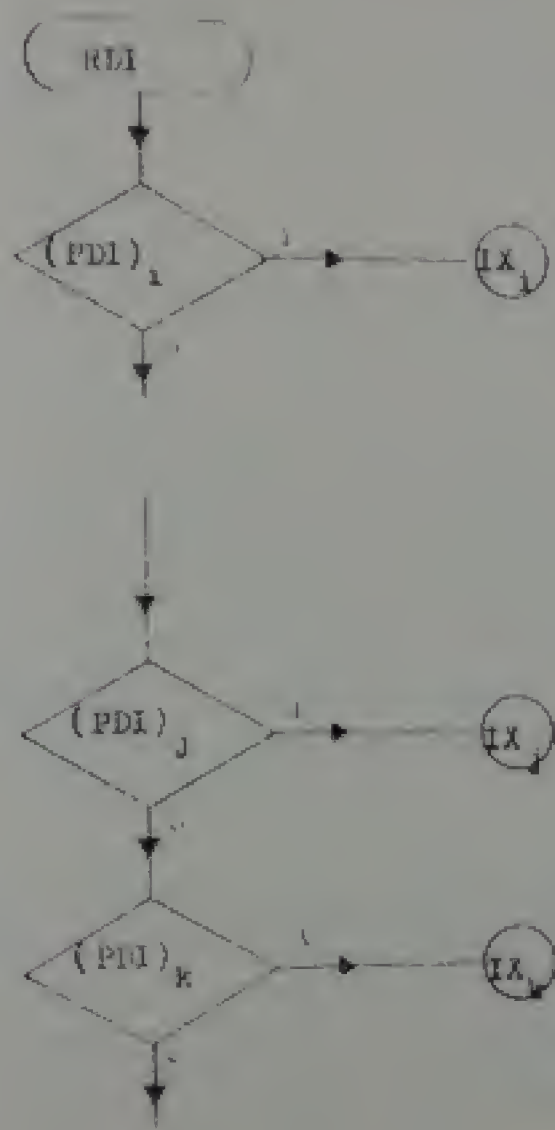
Uma prioridade no atendimento das interrupções é dada pelas interfaces, estabelecendo-se acordo com a ordem em que são realizadas as testes dos sinais PDI pela rotina RDI. Assim, com referência à Fig. 2.2, a interface de número 1 tem maior prioridade que a de número 4, uma vez que há período de interrupção originado na interface 4 só após a ocorrência quando não houver período de interrupção da interface 1.

"Atender" uma interrupção de interface 1 foi utilizado com o sentido de produzir um desvio programado para o processador que se encontra na posição INÍCIO (2.2).

## 2.3 - INTERUPÇÕES SÍNCRONAS E ASSÍNCRONAS

Uma interrupção produzida por uma instrução de máquina é denominada INTERUPÇÃO SÍNCRONA ou PROGRAMADA. Para tipo de

THE CASE OF TWO-TIME INTERDEPENDENCE





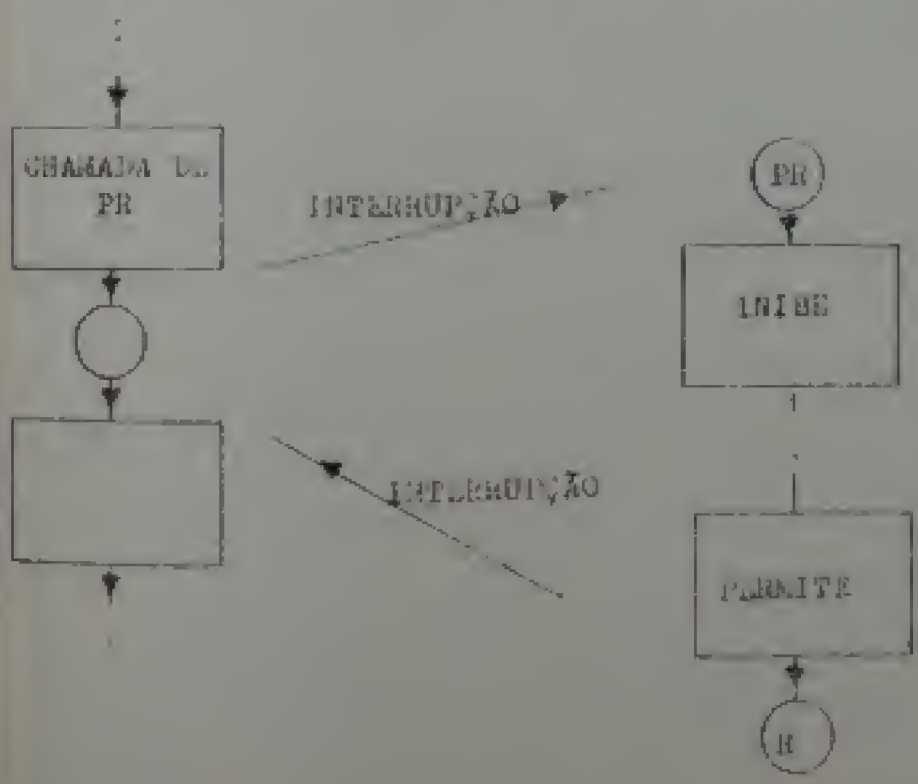


operações de modo que estas interrupções poderiam ocorrer em qualquer instante de determinadas condições não definidas, a serem determinadas durante sua execução, não sendo possível prever exatamente em quais instantes estas irão ocorrer. Devido deste deste fator, o caráter asintótico deste tipo de interrupção.

### 3.4 - PRIMITIVOS E PROCESSOS

Conforme observado no início deste capítulo, o fato de uma instrução de máquina não ser interrompível, permite considerar todas as ações tomadas pela D.C.P. na execução da instrução como sendo um PRIMITIVO. Ainda, pelo que foi visto nos itens anteriores, uma sequência de instruções realizadas pela P.C.P. no estado INTERRUPTIDO, também poderá ser considerado um PRIMITIVO, uma vez que neste estado a D.C.P. não é interrompível.

Entretanto, é necessário considerar um aspecto importante na programação de primitivos. Por exemplo, considerando-se um trecho de programa ou uma rotina PR composta por uma série de instruções que se inicia pela instrução "INIBE INTERRUPTO" ( (PR) = 0 ) e termina pela instrução "PERMITE INTERRUPTO" ( (PR) = 1 ), essa subrotina não implementa um primitivo devido







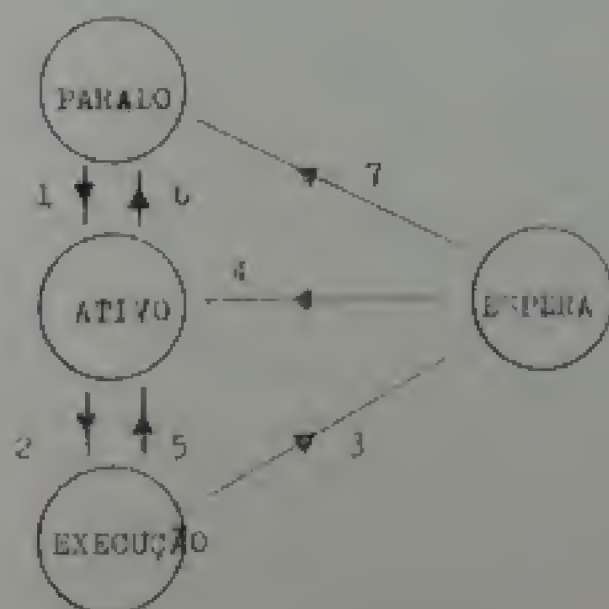
## 2.5 - ESTADOS DE UM PROCESSO

Logo que um processo é identificado, este permanece no estado PARADO. Neste estado o processo não pode utilizar a U.C.P., ou seja, a U.C.P. não irá executar o programa associado a este processo sendo que o único recurso a este atribuído, será a porção de memória ocupada pelo programa.

Somente a partir do estado ATIVO o processo terá condições de utilizar a U.C.P. para a execução de suas rotinas. Toda vez que isto ocorrer, o processo passará para o estado de EXECUÇÃO.

Nos intervalos de tempo em que por qualquer motivo a execução do processo está suspensa, este assume o estado de ESPERA.

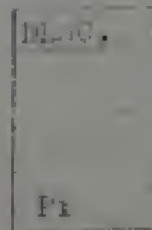
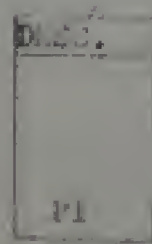
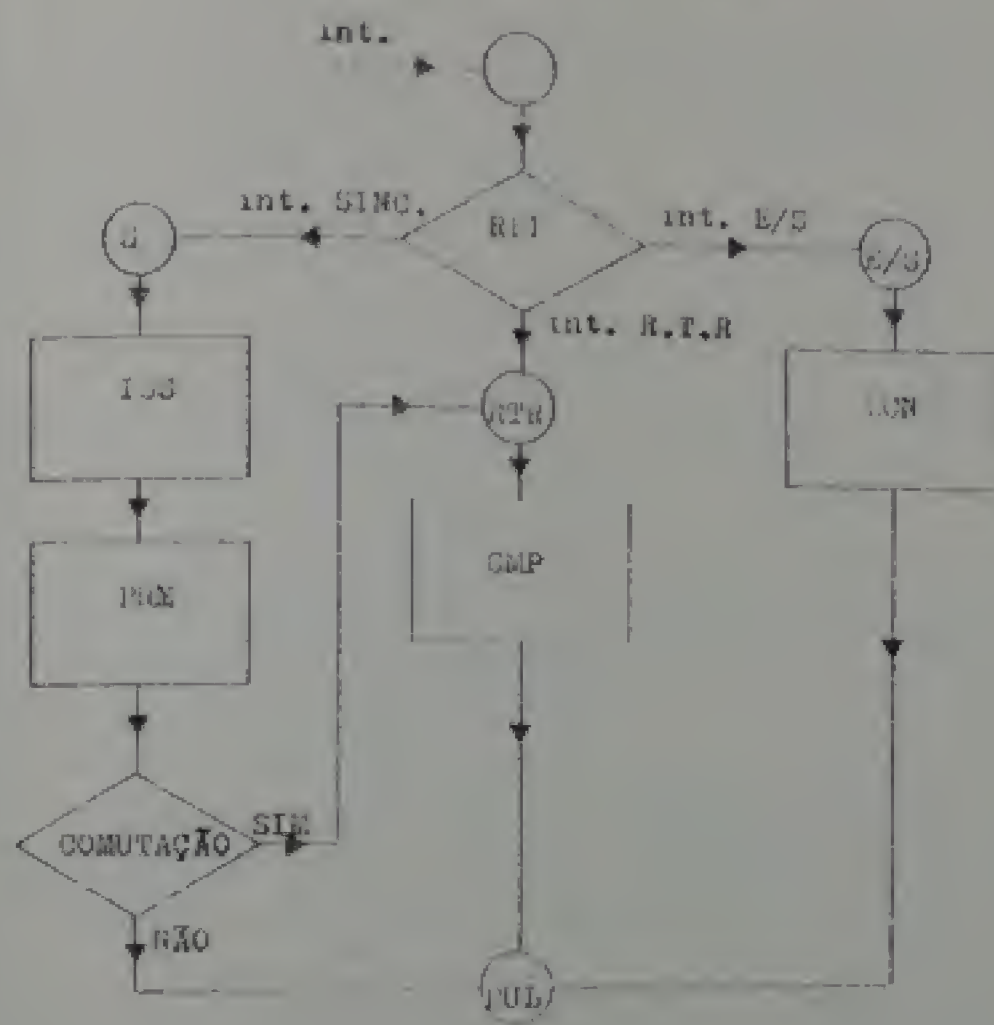
A figura abaixo ilustra esses estados bem como as transições possíveis entre eles.



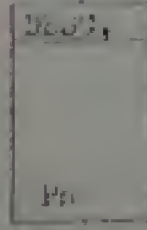
A complexidade dos estados bem como as transições que podem ser realizadas entre eles, decorrem do fato de se admitir a existência de vários processos simultâneos, competindo com as limitações recursos disponíveis no sistema, bem como das interações que possam existir entre eles.







int. SINC.  
int. E/S  
int. R.T.R







## 2.6.1 - ROTINA DE DESACERTAMENTO DE INTERRUPÇÕES (RD1)

Conforme mencionado anteriormente, o RD1 estabelece o ponto de acesso ao MÓDULO da sistema.

A posição do rotulo RD1 (Fig. 2.54) é alcançada por meio de interrupções, devendo ocupar portanto as posições de endereço da memória.

As interrupções de número 1 a 5 foram atribuídas às funções de produzir as interrupções síncronas e as interrupções de RELÓGIO DE TEMPO REAL (2,4,6,8) respectivamente, sendo consideradas as de maior prioridade pelo MÓDULO. O tratamento dessas interrupções é controlado pelas rotinas 155 e 156.

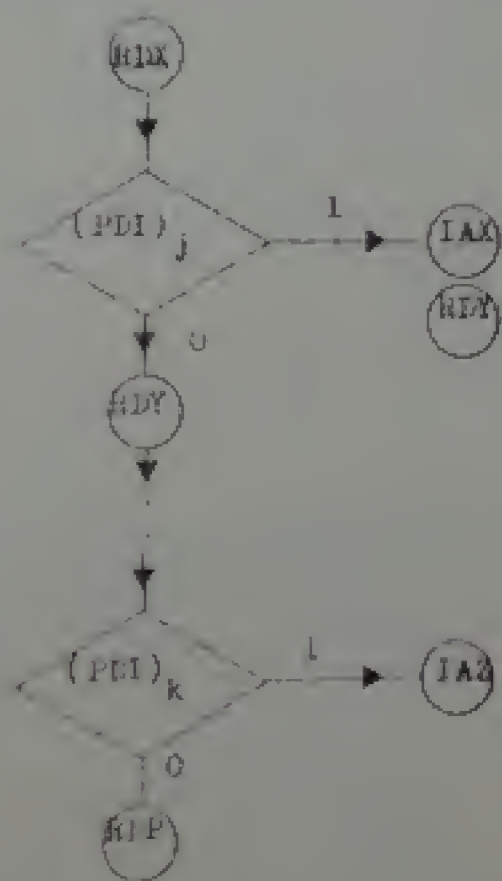
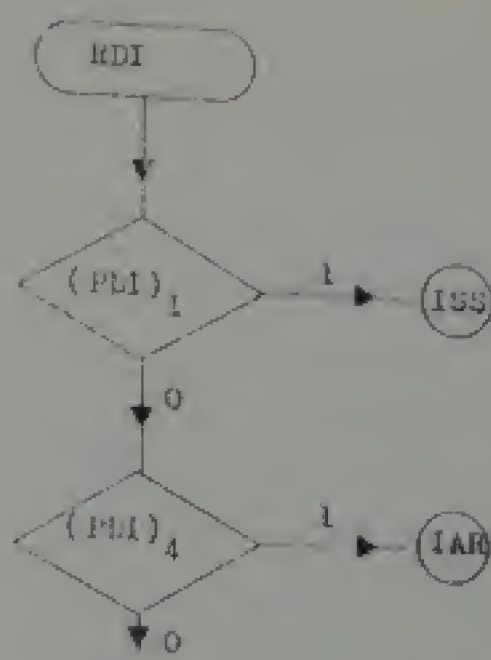
As demais interrupções, até ao número 14, estão ligadas aos dispositivos de ENTRADA e SAÍDA.

No caso de uma determinada interrupção ter tomado a prioridade de se rotular cada teste (RD1) a fim de facilitar, em primeiro lugar, ignorar o resultado de testes (1) quando a interrupção é uma está concluída submetendo-se IAX por RDY, ou seja procurando para o teste da próxima interrupção. Em segundo lugar a troca de prioridades entre duas interrupções por 1 e 2 poder ser realizada trocando-se IAX por IAX', a instrução (P01) e por (P014) e vice-versa.

As interrupções produzidas pelo painel (BOTÃO DE INTERRUPÇÃO DO PAINEL (1)) tem a menor prioridade, sendo tratada por rotina de rotulo R.D.P. Sincronismo com o teste específico para cada tipo de interrupção, de modo que a sua ocorrência é feita por sucessos.

## 2.6.2 - ROTINA DE EXATAMENTO DE INTERRUPÇÕES SÍNCRONAS (155)

A rotina 155 compõe-se de um conjunto de subrotinas que implementam um determinado conjunto de prioridades.







### 2.6.2.2 - ESTRUTURA DE DADOS DA ROTINA ISB

A rotina ISB contém um conjunto de dados estruturados, com o propósito de descrever o andamento de cada processo em andamento no sistema.

A cada processo está associado um DESCRITOR, que é constituído por uma sequência de 10 palavras consecutivas, contendo o "STATUS" do processo e outras informações relativas à respeito do processo.

O "STATUS" de um processo é igual ao "STATUS" da P.C.P. em cada instante, quando este estiver em EXECUÇÃO. Compreende os valores do ACUMULADOR, EXTENSÃO, ÍNDICE, CONTADOR DE INSTRUÇÕES ( $CI_i$ ), e os sinais de TRANSFERÊNCIA ( $CT_i$ ), e "VAL-DM" ( $V$ ), conforme a referência (1).

Quando ocorrer uma criação de processo, o "STATUS" da P.C.P. é armazenado no DESCRITOR do processo que sofrerá a interrupção, enquanto que, nesta ocasião, o "VAL-DM" contido no DESCRITOR do novo processo é ser executado.

Estes DESCRITORES ( $1 \leq i \leq 10$ ) estão localizados nas 10 posições consecutivas que antecedem a primeira palavra do programa associado ao processo e contém:

$ACC$	: ACUMULADOR
$EXT$	: EXTENSÃO
$IND$	: ÍNDICE
$CI_i$	: SINALS ( $1 \leq i \leq 10$ ), VALOR de CONTADOR DE INSTRUÇÕES ( $0 \leq i \leq 10$ ).
$CT_i$	: CONTADOR DE INSTRUÇÕES ( $1 \leq i \leq 10$ ), SINALS ( $0 \leq i \leq 10$ ).
$AD_i$	: DIMENSÃO EM NÚMERO DE PALAVRAS DO PROGRAMA ( $1 \leq i \leq 10$ ).
$DM$	: DIMENSÃO EM NÚMERO DE PALAVRAS DO PROGRAMA ( $0 \leq i \leq 10$ ).

	TEN	TIN	TDL	TDC
0 →	PLP	PEL		
PEL →	EST.	INL	PNC	DP <sub>1</sub> DP <sub>0</sub>

DESCRITOR						PROGRAMA	
ACC	EXT	INL	TV	CI <sub>1</sub>	CI <sub>0</sub>		

Os 5 bits mais significativos de  $XD_1$  podem ser usados para um outro propósito qualquer.

A cada processo em andamento está associado um número de identificação entre 1 e 15. Esta identificação é utilizada no índice correspondente a uma linha da TABELA DE DESCRIÇÃO DOS PROCESSOS (TDP).

A Tabela TDP (TDE - 2.5) contém quatro colunas denominadas por IDS, IBS, PDL e PDR, sendo cada elemento destas colunas constituído por uma palavra (8 bits), em número de 15 elementos por coluna.

#### 1) COLUMNA IDS

A coluna IDS contém informações relacionadas com o TADO, codificadas da seguinte forma:

7	6	5	4	3	0
A	E	P		ISL	

A = 1 : PROCESSO ATIVO

E = 1 : PROCESSO EM ESPERA

P = 1 : PROCESSO PARADO

O número máximo de processos simultâneos (PS) em execução depende da velocidade e extensão de memória ocupada pela Tabela TDP em relação ao total de memória disponível (TMR) e por tecnologia de tecnologia de hardware.

A possibilidade de atribuir a execução simultânea de processos, a que era processada sequencialmente, tendo em conta a sua prioridade na TDP, permite reduzir a necessidade da implementação de um esquema de rutinas e respectiva de elementos desta Tabela.



Este esquema está baseado em duas listas, cada uma com percorrendo as linhas ocupadas e a outra as linhas vazias de TDP.

Para este efeito, os quatro bits menos significativos de cada elemento da coluna TDP, TML e dos indicados, com INDICADOR na lista, possibilitando referir-se a qualquer linha da tabela.

Estas listas encadeadas são implementadas utilizando-se as variáveis PIP e PDL indicando a primeira linha da lista de processos e a primeira linha da lista vazia respectivamente e o encadeamento continuando pelos valores de ISI. Em particular ISI = 0 determina o final de uma lista.

De um modo geral, sempre que uma lista contiver encadeamento, este é realizado utilizando-se a primeira linha da tabela com os quatro bits mais significativos (PIP) indicando o primeiro elemento utilizado e os quatro bits menos significativos (PDL), o primeiro elemento livre.

Com esta padronização, uma rotina subrotina (PQTI) poderá realizar as operações de inserção e retirada de elementos de uma tabela encadeada bastando que se forneça o endereço do início da tabela.

Uma situação particular destas listas está representada na Fig. 2.6.

#### 4.3 COLUMNA TDP

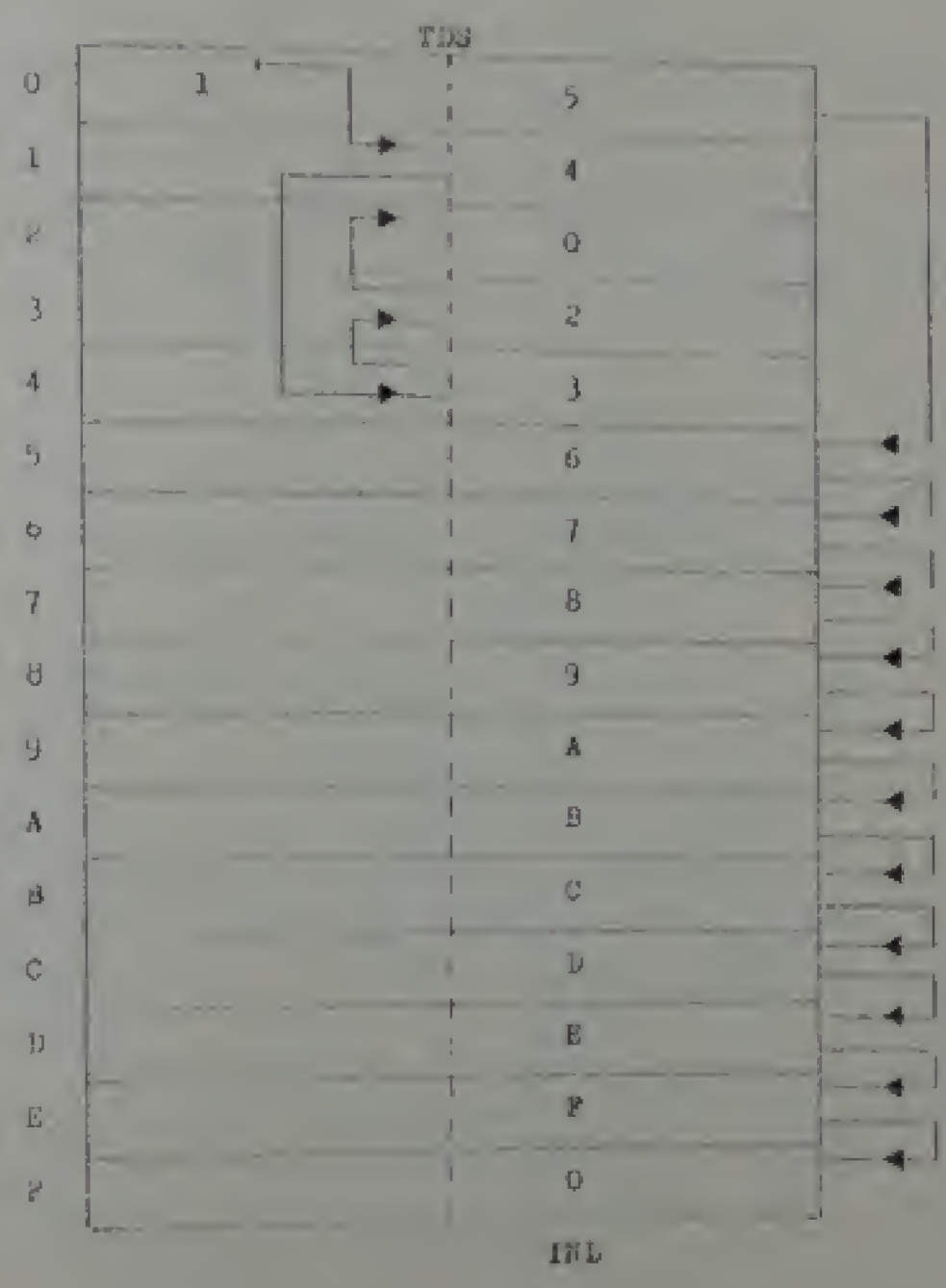
A cada processo associado um código de 4 a 255 indicando por um elemento da coluna TDP.

Este código concentra-se na maneira pela qual um processo pode referenciar um outro e fornece proteção para que a identificação única de cada processo possa sofrer alterações durante a execução.

#### 4.4 COLUMNA TDI e TDU

Os elementos das colunas TDI e TDU contêm um apontador para o INDICADOR de processo e o elemento de controle de estado.





bits é codificado em TDL e posto à disposição da unidade de TDL.

Os quatro bits são então decodificados de TDL, sendo utilizados como complementos de cada bit da TDL que seja decodificada (Fig. 2.6.2).



PRS : Índice (de 0 a 15) de processo subordinado

DP<sub>1</sub>, DP<sub>0</sub> : Apontador para o DESTINATÓRIO do processo.

#### 2.6.2.4 - ESTRUTURA DA ROTINA TSC

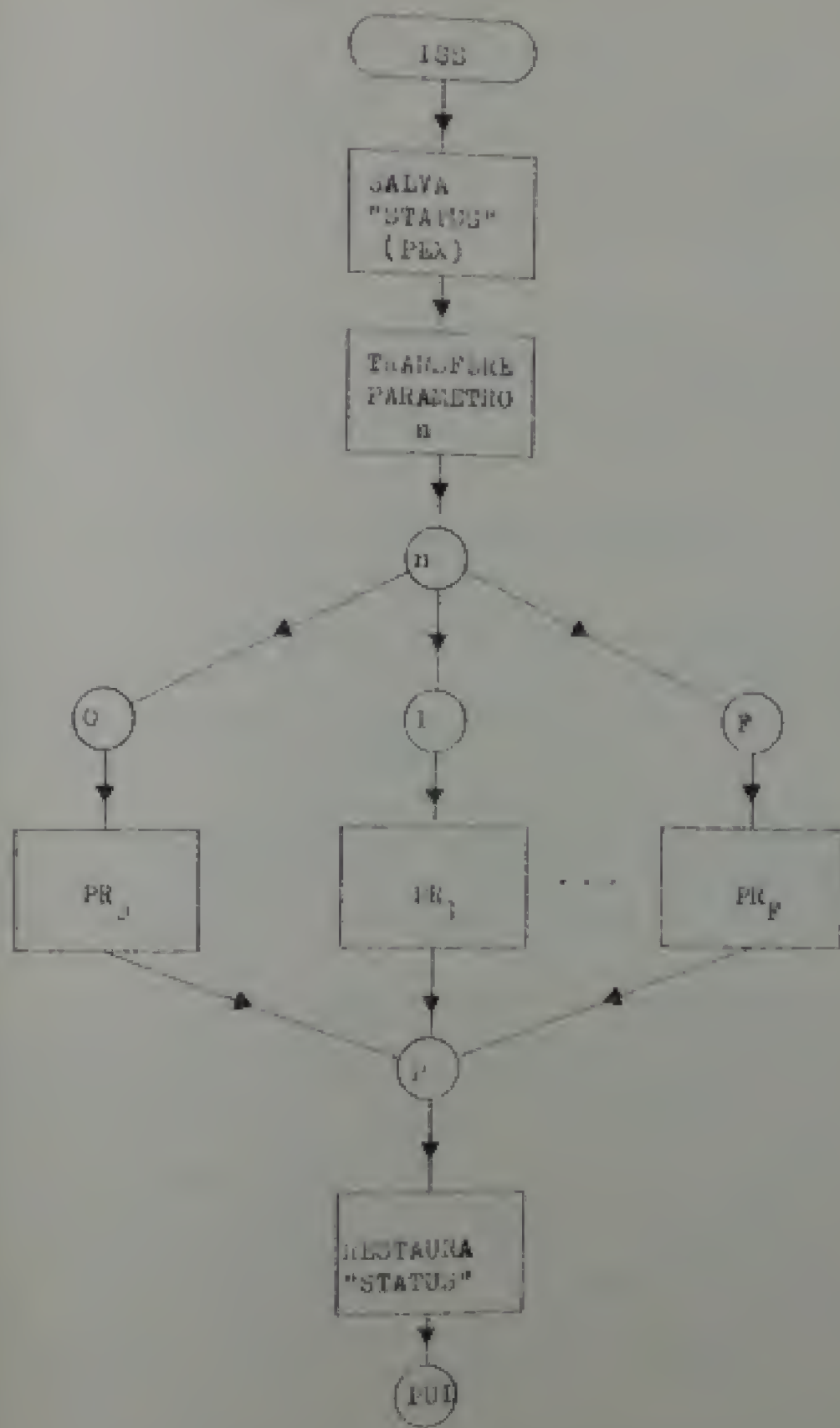
A Fig. 2.7 ilustra o diagrama da rotina TSC.

Inicialmente, após a execução da instrução GDB continua-se pelo processo em EXECUÇÃO, identificando pela variável PR<sub>0</sub> o "STATUS" desse processo e salva-se em algumas variáveis temporárias.

O endereço dos parâmetros é obtido, o endereço é posto em conteúdos das posições 0 a 3 do registrador, após o que o primeiro endereço (n<sub>1</sub>) é obtido. O valor n<sub>1</sub> identifica o primitivo a ser utilizado e, como esse valor encontra-se valor máximo (MPR) o primitivo é o MPR e é utilizado.

Após a identificação do primitivo através de n<sub>1</sub>, é utilizado um desvio para as subrotinas que executam o particular do primitivo PR<sub>0</sub> a PR<sub>n</sub>, dando d = SPP.

A execução de cada primitivo termina após a realização do "AJUSTE" do D.C.D. por meio das variáveis temporárias. A obtenção dos parâmetros necessários às instruções S, dependendo da instrução, é fornecida por meio da instrução PPR, a D.C.D. realiza a "CORREÇÃO" do estado, retornando para o estado normal.





os DESCRIPTORES de processos também são atualizados quando algum primitivo ocasiona uma comutação de processo. Neste caso, o DESCRIPTOR do processo interrompido é atualizado com os valores contidos nas variáveis temporárias e estas recebem os valores de DESCRIPTOR do novo processo que deverá ser executado. Essas operações são realizadas pela subrotina *process* tipo de comutação de processos.

Os parâmetros constantes da sequência de chamada da subrotina *CSP*,  $n_1, n_2, \dots, n_k$  são transferidos pela subrotina *TII* e estão sob controle do particular primitivo associado através de  $n_1$ .

### 2.6.1 - TRATAMENTO DAS INTERRUPTÕES ASSÍNCRONAS

O tratamento das interrupções assíncronas, em geral provocadas por interfaces ligadas a dispositivos de entrada e saída, não é necessariamente padrão, dependendo da natureza do dispositivo conectado à cada interface.

Vendo os dispositivos atualmente disponíveis no "PARTIAL FEIO", estas interfaces foram subdivididas em três grupos de acordo com algumas características comuns.

O primeiro grupo engloba os dispositivos de entrada e saída conectados a interfaces padrão de "8 bits" que podem provocar interrupções na transferência de cada CARÁCTER de 8 bits entre a memória e o dispositivo no caso de uma saída ou entre o dispositivo e a memória, no caso de um entrada. A maioria das interfaces atualmente disponíveis é deste tipo e controla as operações de LECTURA DE FITA DE PAPEL, PERFORADORA DE FITA DE PAPEL, TELEIMPRESSORA (TTY), RELEVO LECTOR DE CARTÕES e IMPRESSORA DE LINHAS. A especificação de cada dispositivo utilizado no Sistema Básico de Controle aparece no Capítulo 4.

No segundo grupo, entram as interfaces ligadas a dispositivos caracterizados como por exemplo DISCO MAGNÉTICO, FITAS MAGNÉTICAS nos quais a transferência de dados é quantificada de acordo



em velocidades bem altas que os anteriores exigem velocidades capazes de produzir aquecimento e degradação principal nos dispositivos da B.C.P. e proporemos um tratamento distinto das interrupções para cada caso. Neste grupo foram incluídas ainda algumas interrupções ligadas a CONVERSORES tipo ANALÓGICO/DIGITAL ou DIGITAL/DIGITAL ou ainda MULTIPLEXADORES de tipo de dispositivo.

Finalmente, o terceiro grupo vai reunir as interrupções que interferem, por meio de interrupções, diretamente com o NUCLEO como é o caso da "Interface S" (2.1.1) e uma interface chamada REGÍSTRIO DE TEMPO REAL capaz de produzir interrupções periódicas. No caso, somente esta última é considerada geradora de interrupções assíncronas conforme definido em 2.1.1. Neste trabalho apenas para os dispositivos do primeiro grupo foram implementados rotinas de tratamento, esquematizadas em 2.6.2.1.

2.6.2 - AÇÃOAMENTO DE PROCESSOS

O açãoamento de um processo consiste em colocar o mesmo no estado de "EXECUÇÃO", ou em outras palavras fornecer a B.C.P. por um mecanismo apropriado um sinal de início de execução durante um determinado intervalo de tempo.

Durante a execução de um processo, tendo esta sido iniciada pelo usuário ou pela B.C.P., a possibilidade de sua interrupção pelo sistema somente ocorre quando esta foi interrompida.

Uma interrupção pode ser provocada tanto por interrupções de processo em execução, através de uma interrupção assíncrona por eventos externos como no caso de uma interrupção assíncrona. A primeira hipótese é a mais visível, podendo ocorrer casos em que um processo iniciado na B.C.P. por tempos indefinidos, por exemplo, por um sinal funcionamento do processo.

Com a possibilidade de paratizar um determinado sistema de B.C.P. surge a possibilidade de açãoamento de processos, podendo ser pelo qual cada processo é executado periodicamente em intervalos de tempo fixos e bem definidos, ou qual este tempo da B.C.P.

Este esquema, também chamado "ROUND-ROBIN" (A) é implementado utilizando-se o recurso da interface RELÓGIO DE TEMPO REAL.

O RELÓGIO DE TEMPO REAL, conectado na interface de número 6 e com o segundo nível de prioridade (2.2), é capaz de produzir interrupções periódicas, o período pode ser programado por meio de interrupções especiais para a interface, podendo ser fixado entre 1 milissegundo e 1000 segundos.

As interrupções assíncronas produzidas pelo RELÓGIO DE TEMPO REAL (RTR) são atendidas pela rotina 158. Esta rotina, esquematizada na Fig. 2.10, após cada interrupção, atualiza o descritor do processo interrompido, da mesma forma que realizada pela rotina 155 e passa a obter o próximo processo ativo.

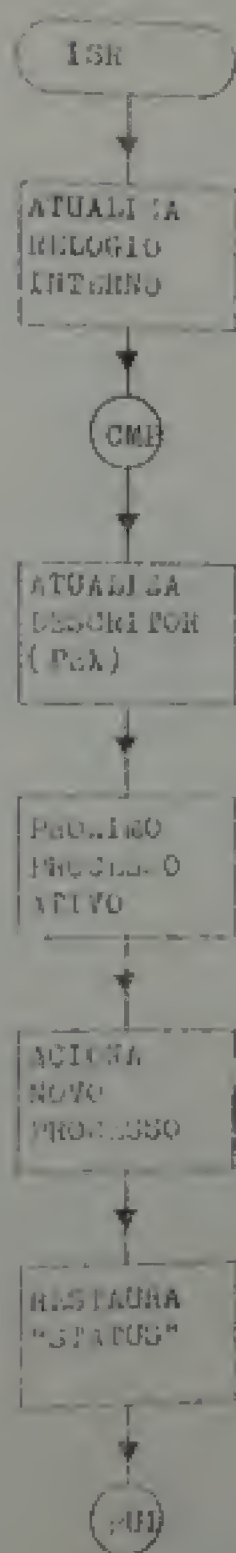
A busca do próximo processo ativo é realizada explorando-se o encadeamento de todos os processos em andamento, disponível na TABELA DE DESCRIÇÃO DOS PROCESSOS (TDP).

A partir do posição indicada pela variável PIX, correção pendente ao processo interrompido, a lista da coluna PBS é percorrida e o primeiro processo encontrado no estado ATIVO é escolhido.

A variável PIX é atualizada conforme o valor obtido no bloco, colocando o processo correspondente no estado de "EXECUÇÃO". Uma operação similar é realizada pela subrotina "PROXIMO PROCESSO ATIVO" (PAV).

Quando o novo valor de PIX, o pedido de interrupção do RTR, é atualizado ( $INTPR = 0$ ), uma nova interrupção é permitida e por isso o novo processo é alocado da mesma forma que na rotina 158.

A fixação do período de tempo atribuído a cada processo não é definitiva, sendo necessário observar o tempo total do sistema para cada valor experimental e tal de se obter um valor ótimo para cada conjunto de parâmetros experimentais.





Entretanto, um valor mínimo pode ser avaliado, considerando-se o tempo utilizado pela rotina  $100$ . Para essa implementação, este tempo foi avaliado em cerca de  $200\mu$ , de modo que o intervalo de tempo não deve ser menor que este. Na perspectiva especial (PRI), garante que o "Time Slice" está sendo dado de acordo com as necessidades.

#### 2.6.3.1 - PRIMITIVOS RELACIONADOS COM O RELÓGIO DE TEMPO REAL

Três primitivos foram implementados a fim de possibilitar o controle do RELÓGIO DE TEMPO REAL e do variável RELÓGIO INTERNO.

##### 1) PROGRAMA RELÓGIO DE TEMPO REAL (PRI)

Memoriamente, PRI(n) programa o RELÓGIO para pedir interrupções periódicas, com períodos de  $10^0$  ms,  $n = 0, 1, 2, 3$ . A cada uma das primitivas  $n$  a seguinte:

CSP

n

n

onde  $n$  identifica esse primitivo.

##### 2) RF RELÓGIO INTERNO (IRI)

O variável RELÓGIO INTERNO (IRI) controlada por três posições consecutivas pode ser lida pelo primitivo IRI ( $i_0, i_1, i_2$ ) onde  $i_0, i_1, i_2$  são três posições que separam a instrução IRI, onde estão armazenados o valor do IRI.

CSP

i

$i_0$

$i_1$

$i_2$

A instrução IRI é implementada a seguir (IRI) por um micro T.M.

# 1. ATENÇÃO AO LÓGICO INTERNO (ARL)

Analogamente a RPL, o primitivo ARL  $(I_0, I_1, I_2)^3$  (analogia) tem os valores consecutivos  $I_0, I_1, I_2$ . Para as portadoras consecutivas da saída a variável RPL.

C-P

2

$I_0$

$I_1$

$I_2$

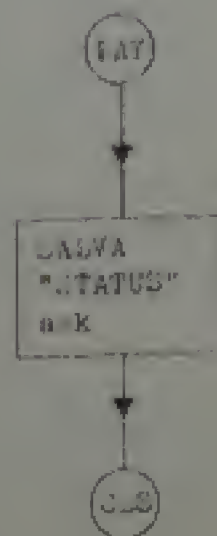
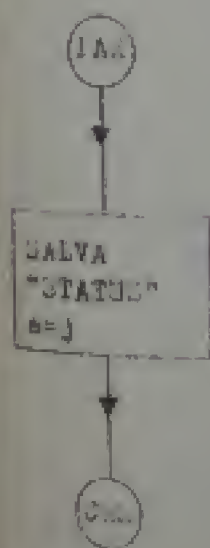
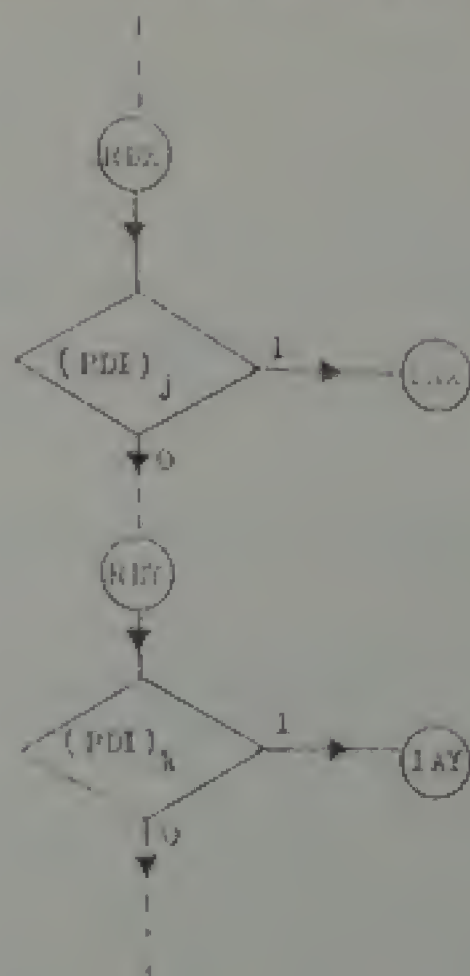
## 2.6.5 - CANAL CONCENTRADOR DE ENTRADA E SAÍDA

O termo CANAL CONCENTRADOR DE ENTRADA E SAÍDA foi utilizado para referir-se ao conjunto de subcanais a serem usados de modo de NÚMERO de destino ao tratamento das interrupções e controle de das interrupções por meio de 3 bits, conforme mencionado em 2.6.4. Uma analogia com os CANAIS CONCENTRADOR DE ENTRADA E SAÍDA implementados em "HARDWARE" (11) é aqui realizada, tendo-se por objetivo a simulação pelo RPL de um processador de entrada e saída para os dispositivos mencionados no primeiro grupo em 2.6.1.

As interrupções produzidas por uma interface ligada ao canal concentrador são identificadas pela rotina RPL, conforme descrito anteriormente, e controlada a nível de RPL no sub-devisivo adicional é feita para uma subrotina de interrupções (IAR) com endereços distintos para cada rotina. Assim, para o caso de interrupções realizadas no canal de interrupções, o RPL.

### 2.6.5.1 - SALVAR O "STATUS" DA D.C.P.

O "STATUS" da D.C.P. nos casos de interrupções, é armazenado no valor de COMPROMISSO DE INTERUPÇÕES que é armazenado automaticamente nos registros  $I_0, I_1, I_2$  e utilizado para o controle de interrupções. O endereço da subrotina RPL.





## 3.1 IDENTIFICAÇÃO DE DISPOSITIVOS

Uma vez que o controle das operações de identificação é semelhante, este é realizado por uma única rotina chamada CONTROLO DE ENTRADA E SAÍDA (CES). Após o sucesso do "STATUS" da rotina de identificação correspondente a programação, com o número de interface correspondente, o dispositivo realiza um teste funcional para a rotina CES.

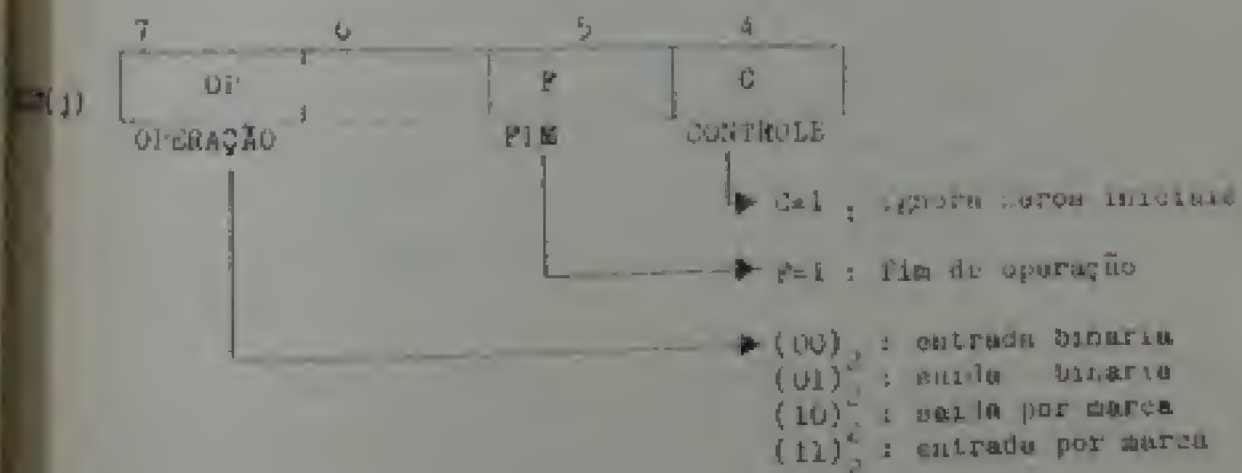
A Fig. 2.9 esquematiza esses procedimentos.

A rotina CES utiliza uma tabela chamada TABELA DE FICHAS DOS DISPOSITIVOS (FID) (Fig. 2.10) onde cada linha está associada à interface. A coluna FIC corresponde ao conteúdo de caracteres (CONT) a serem transferidos enquanto que a coluna FIB contém a parte menos significativa do endereço  $16_{16}$  de primeira porta da memória principal de onde os dados são transferidos para todos os caracteres. A coluna FIB contém nos bits 0 a 7 a parte mais significativa do endereço  $16_{16}$  e nos bits 8 a 15 de 4 a 7 são reservados para a descrição do tipo de operação sendo realizada bem como o estado da rotina RES.

O bit "FIM DE OPERAÇÃO" indica quando "1" que a última operação de interface foi realizada. A rotina de disposições espera de realizar tanto operações de entrada como de saída de caracteres, caso de uma decompressão e copia e quando a operação de transferência é realizada. A rotina de "TRANSFERÊNCIA" de uma transferência é realizada utilizando-se caracteres armazenados em "ASCII" (AMERICAN STANDARD FOR INFORMATION) por exemplo, a descrição da operação pode ser realizada pela descrição de um caractere tipo bit, chamado MARCA.

Cada operação é gerenciada pelo bit "TRANSFERÊNCIA" (FIM MARCA), sendo que a marca é o bit menos significativo do código de código (CY)  $16_{16}$ .

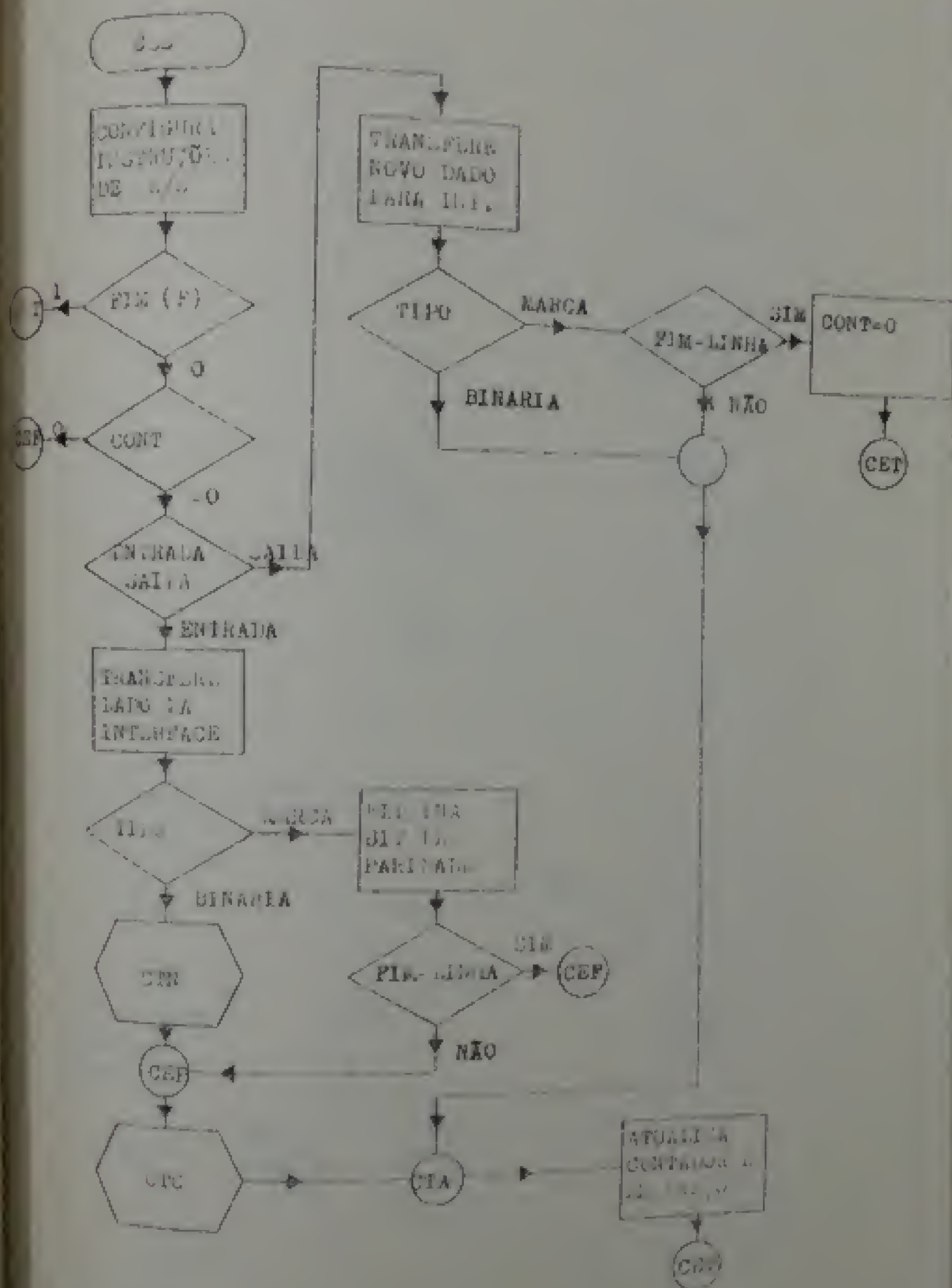
Determinados eventos ocorridos no funcionamento de uma rotina, por exemplo, a ocorrência de um erro, são armazenados em uma rotina de erro.

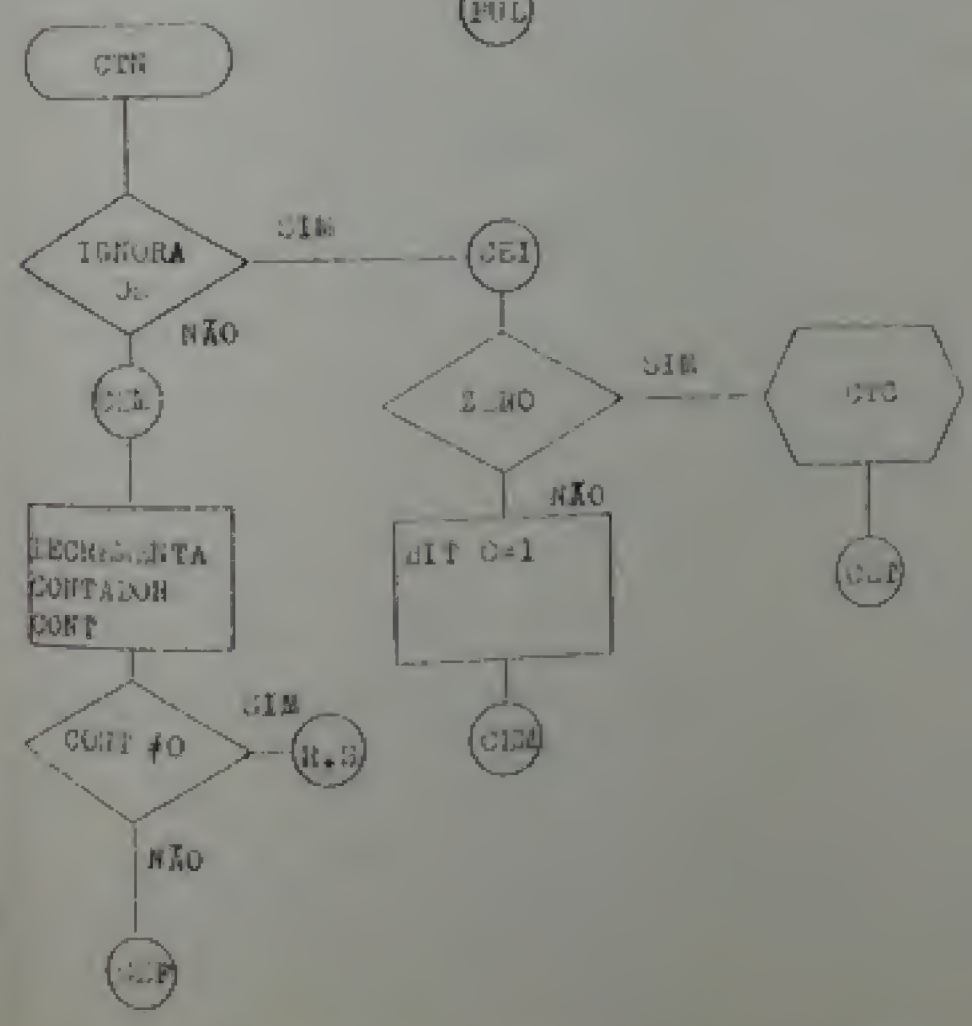
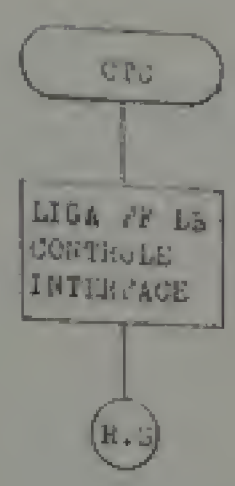
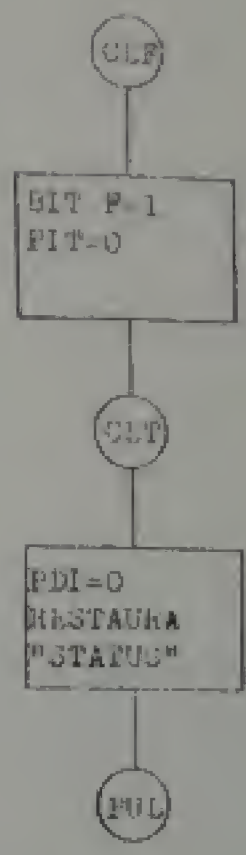
[illegible]

A continuidade da operação realizada pela rede elétrica, (iii) consistir de uma sequência de testes de carga de estado de tensão correspondente à intensidade que provocou a interrupção, com a verificação de novos dados em função da duração da MARCHA, tendo que de um fim de operação, e finalmente a restauração do "STATUS" da UEP e transição desta para o estado NORMAL a cada interrupção.

Uma vez que a ULS é comum a várias interfaces, é necessário que as instruções de entrada e saída sejam montadas a cada interrupção, de acordo com o número da interface sendo tratada. Além disso, a necessidade de um grande número de testes, o salvamento e recuperação do "STATUS" da ULS, podem tornar o atendimento dessas interrupções demandando o suficiente para prejudicar o desempenho do sistema quando vários dispositivos forem operando simultaneamente. Entretanto, a economia de memória proporcionada por este esquema podem tornar essa deficiência suportável em uma versão mais simples do sistema operacional. Porventura pode-se considerar substituir-se a ULS por rotinas especializadas para cada interface que exigirá uma velocidade muito inferior no tratamento de uma interrupção.







os códigos de controle são os  $1000_{16}$ ,  $0001_{16}$  e  $0001_{16}$  respectivamente, significando que os demais parâmetros têm função de especial para cada um deles.

#### a) 1º ESTADO DO DISPOSITIVO (111)

O primitivo 111 (1), onde 1 indica o valor da função de controle e adicionando da seguinte forma:

ESP  
S  
I  
B

Após a execução do primitivo, o valor do parâmetro 1 é armazenado como o conteúdo dos bits EST (3) (12, 2-10) e devendo ser interpretados da seguinte forma:

- 1 = 1 : Dispositivo fora de operação ou transferência realizada com erro.
- 1 = 1 : Última operação realizada e dispositivo desocupado.
- ou : Operações realizadas.

#### b) 2º ESTADO DO DISPOSITIVO (111)

Este primitivo AES (14) é executado com parâmetro de controle 111 e indica a linha de tabela de controle do dispositivo e com os bits de estado EST (12), onde os bits  $h_1$  (1),  $h_2$  (2) e contador  $h_3$  (3) são os bits de controle  $h_1 = 1$ ,  $h_2 = 1$  e  $h_3 = 1$  respectivamente.



OSP

02.

0

1

1

1

1

1

1

Além do preenchimento da tabela nenhuma outra providência é necessária para a execução desse primitivo.

#### 1) INICIA ENTRADA E SAÍDA (IES)

IES (1) inicia condicionalmente a operação de entrada e saída nos dispositivos  $i$ , de acordo com as informações disponíveis na TABELA DE DESCRITORES DOS DISPOSITIVOS correspondentes a esse dispositivo  $i$ .

OSP

1

1

A condição necessária para que as estruturas de entrada e saída sejam iniciadas no dispositivo  $i$  que o bit  $F_{di}$  da Tabela Operações seja igual a 0 (zero), de outra forma, é igual a 1 (um) a execução do primitivo é terminada. O bit  $i$  determina, se igual a 1, que em uma entrada de dados, os caracteres "moleculares" sejam ignorados.

Além dos dois primitivos, a alteração da prioridade no atendimento das interrupções pode ser conduzida por meio do primitivo especial "TROCA DE PRIORIDADES" TEP (1, 1) 17, 1).

Este primitivo é codificado como:

OSP

10010

1

1

615  
A execução desta causa usa uma troca de posições dos pedidos de pedidos de interrupção das interfaces de números 1 e 1, realizada no "ROTINA DE DESEMPENHO DE INTERFERÊNCIAS" (RDI).

O uso deste primitivo permite que se aplique a influência da prioridade de uma interface na velocidade de operação dos dispositivos de entrada e saída quando utilizados simultaneamente.

#### 2.6.6 - COMUNICAÇÃO ENTRE PROCESSOS

A comunicação entre os processos se realiza mediante uma troca de MESSAGES.

Uma mensagem será definida como sendo um conjunto de palavras consecutivas de número variável indicadas por endereço disponível no NÚCLEO. Estas MESSAGES ocupam posições da memória reservadas por um processo, sendo que o controle e interpretação de cada MESSAGES está a cargo do respectivo processo.

AO NÚCLEO está atribuída a função de transmitir essas MESSAGES por meio de manipulação dos endereços das MESSAGES.

Para esse efeito o NÚCLEO mantém uma lista de endereços de MESSAGES, estando em cada um dos quatro quadrantes as identificações dos processos REMITENTE (R), e DESTINATÁRIO (D), com forma a Fig. 2.12.

A TABELA DE DESCRIPTORES DE MESSAGES é constituída por 15 linhas, possibilitando portanto a manipulação de até 15 mensagens simultâneas, linhas estas encabeçadas por meio de uma lista ligada, cujo primeiro elemento é indicado pela variável PRP, e ainda por uma lista de mensagens vazias encabeçada a partir de uma variável denominada de VZ. A tabela contém quatro bits por linha, os quais são denominados de bits de identificação da coluna TML.

Na coluna TML encontram-se as identificações dos processos envolvidos com a mensagem indicada pelo endereço PRP. A tabela contém 15 linhas, sendo as primeiras 15 linhas TML (4 bits) e as últimas 15 linhas TML (4 bits) e as últimas 15 linhas TML (4 bits).





Este primitivo tem como finalidade, para a troca de MENSAGENS entre os processos, além de fornecer um método de sincronização entre eles.

### 3) ENVIA MENSAGEM (EMS)

O primitivo EMS ( $N$ ,  $T_1$ ,  $T_0$ ), codificado como :

CSP

(9)  $_{16}$

$S$

$T_1$

$T_0$

onde  $T_1$ ,  $T_0$  designa o endereço da área de memória do processador (centro) de uma MENSAGEM a ser enviada ao processo  $N$  (destinatário).

A execução do primitivo se processa, em primeiro lugar, enviando o processo destinatário (o processo  $N$ ) para o estado **VIVO** se este estiver no estado **ESPERA**. No mesmo lugar, os valores de  $T_1$ ,  $T_0$  são como descrito através de  $N$ , e  $S$  obtido através de  $PBS$  (identificador do processo de origem) na linha da primeira linha disponível da TABELA DE DESCRITORES DE MENSAGENS.

Após a execução do EMS, o controle do C.C.P. retorna ao processo remittente.

Uma vez o processo identificado por  $N$  não esteja, no momento da linha disponível na tabela, o processo continuará no estado **VIVO**, isto é, o seu "status" é guardado de modo que quando for novamente repetido este mesmo primitivo.

Este é realizado sob condições onde, quando no C.C.P. um ou VARIÁVEIS do processo remittente por causa de execução do primitivo se encontram no estado **VIVO**.



No caso de interrupção, uma mensagem enviada a tempo-  
mundo (isto é, de  $T = 1$ ) de processo.

II ACORDA MENSAGEM (AMM)

O primitivo AMM ( $CE_{15}$ ,  $CE_{16}$ ) significa o seguinte:

ESP

(B) 16

$T_1$

$T_n$

é produzido de modo análogo aos primitivos AMM, exceto que a MES-  
SAGEM recebida pelo seu lado enviada por qualquer processo em an-  
damento no sistema. Para isso, no caso de AMM, somente o pro-  
cesso destinatário (PI) é utilizado na tabela TSP.

#### 2.6.6.2 - SÍNCRONIZAÇÃO DE PROCESSOS

Os primitivos TMS, AMS e AMM podem ser utilizados na  
sincronização de dados em mais processos do que dois, que se tem  
alguns estados provocados por estes primitivos. Quando se tem  
um grupo de se controlar os instantes em que tais processos devam  
ser executados em dados por ocorrerem em TSPED, e o tempo de execu-  
ção controlado pelos MENSAGENS trocados entre eles.

A fim, TMS justifica tal evento sendo controlado por um  
processo destinatário entre os EXECUTAD, enquanto que AMS e AMM  
podem provocar o mesmo necessário no tempo de execução.

Quando tais eventos são controlados diretamente com MMS  
TMS, o primitivo AMM ESTADO DE PROCESSO (2.6.6.2) pode ser utili-  
zado com o propósito de sincronização.





Estas listas que implementadas de forma encadenada, nos brindan una tabla T(i) conforme figura 2.14. Cada elemento de lista contiene la identificación de proceso subordinado PSS i, un índice de la próxima elemento de la lista PSS (PSS = 0 termina una lista), y control de esta tabla es regulado por una variable que contiene el índice de lista de elementos libres de la tabla.

A segunda hipótese induzida é a de que todos os recursos autorizados por um processo subvindicado, exceto o U.C.I. e o Saco Paralelo, e retirados pelos respectivos promissores controladores,

## 2.0.2.1 - PRIMITIVELY INDEPENDENT.

Fiz um esboço das indicações para quatro primitivos para o controle de processos.

1) **АВТОМАТИЗИРОВАННОЕ ПРОВЕРЕНИЕ РЕЗУЛЬТАТОВ**

O primitivo ADDITIONAL NOVO PROCESSO, CRR (N<sub>1</sub>, S<sub>0</sub>), é acompanhado dos seguintes parâmetros:

60

9		PHI.
C	PSS	PXS
K		



o processo com endereço  $addr$ , inicialmente o valor "PARADO", se puder ser utilizado a V.A. quando o processo for criado por  $addr$ .

Antes que um processo seja adicionado, se verifica se pertence o processo ao subconjunto de processos a serem adicionados, e se o processo estiver de memória, pertencente ao processo  $addr$ . Uma vez que não se dispõe da prioridade do endereço de memória principal onde cada processo deverá ser armazenado, em função do endereço relativo, um formato especial para a representação dos programas deverá ser utilizado. Estes aspectos são discutidos no capítulo 4.

#### 5.1. ATRIBUIÇÃO DO ESTADO DE UM PROCESSO (AEP)

Um processo CONTROLADOR pode alterar o estado de um processo subordinado SUBORDINADO mediante o uso da primitiva AEP (5.1).

CSP

(1)  $16$

$S$

$E$

onde  $S$  é o código do processo SUBORDINADO e  $E$  o endereço que este possui atualmente.

$E = (70)_{16}$  : PARADO

$E = (60)_{16}$  : EXECUTANDO

$E = (80)_{16}$  : ATIVO

O endereço  $addr$  é armazenado no campo correspondente ao código  $S$  da coluna TPC. Se o processo de código  $S$  não for SUBORDINADO, o processo não poderá ser alterado.

Um processo  $K$ , bem como todos os processos subordinados por ele, são subordinados, são removidos através do primitivo RMP (R).

RMP

$(R)_i$

$N$

onde  $N$  é o código do processo a ser removido.

A remoção de um processo consiste na liberação da identificação associada ao processo e a consequente atualização da TDP com o lançamento da linha contendo a descrição do processo na linha vazia.

Este procedimento é realizado com o auxílio da coluna na base das identificações dos processos "descendentes" do processo removido.

Como nos outros casos, a existência do processo  $N$  ou não autoriza ou não a execução do primitivo RMP, conforme um "ATRIB" no processo emitente.

#### 4. LECTURA DO DESCRIÇÃO

A leitura do DESCRIÇÃO de qualquer processo pode ser realizada mediante o primitivo LDP (L,  $i_1, i_n$ ).

LDP

$(L)_i$

$N$

$i_1$

$i_n$

Por meio deste primitivo, se conhece o DESCRIÇÃO de qualquer processo  $N$  e a identificação por  $i_1, i_n$ , permitindo assim que seja feita a "LEITURA" do processo por meio de suas identidades.

CAPÍTULO I - UM SISTEMA BÁSICO DE CONSELHO  
PARA O "PALINHO" 1010





## 1.

1.

1.



1.

1.

1.

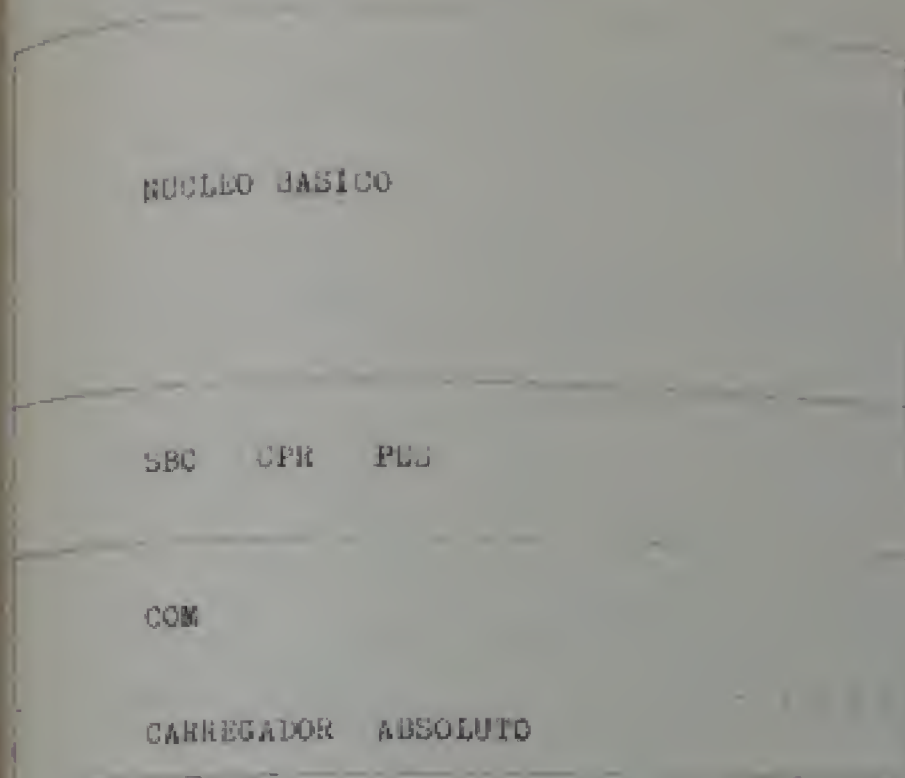


Fig. 1.1

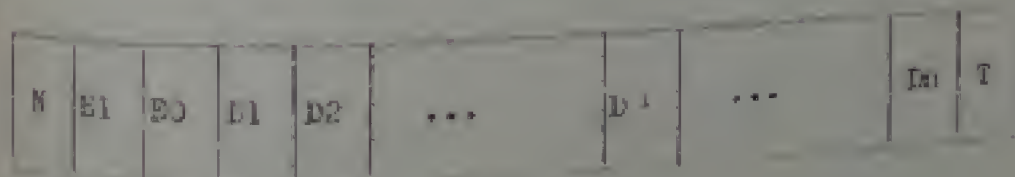


Fig. 1.2



[illegible][illegible]

Estes códigos fazem parte de um arquivo em fita de papel pertencente ao formato ABRAHIM. De este modo, pode-se utilizar o CACHUQUE ABRAHIM nesta operação, depois que a execução de uma base de controle está sendo usada no processo AB.

王. 1997. 1998. 1999. 2000. 2001. 2002. 2003. 2004. 2005. 2006. 2007. 2008. 2009. 2010. 2011. 2012. 2013. 2014. 2015. 2016. 2017. 2018. 2019. 2020. 2021. 2022. 2023. 2024. 2025. 2026. 2027. 2028. 2029. 2030. 2031. 2032. 2033. 2034. 2035. 2036. 2037. 2038. 2039. 2040. 2041. 2042. 2043. 2044. 2045. 2046. 2047. 2048. 2049. 2050. 2051. 2052. 2053. 2054. 2055. 2056. 2057. 2058. 2059. 2060. 2061. 2062. 2063. 2064. 2065. 2066. 2067. 2068. 2069. 2070. 2071. 2072. 2073. 2074. 2075. 2076. 2077. 2078. 2079. 2080. 2081. 2082. 2083. 2084. 2085. 2086. 2087. 2088. 2089. 2090. 2091. 2092. 2093. 2094. 2095. 2096. 2097. 2098. 2099. 2100. 2101. 2102. 2103. 2104. 2105. 2106. 2107. 2108. 2109. 2110. 2111. 2112. 2113. 2114. 2115. 2116. 2117. 2118. 2119. 2120. 2121. 2122. 2123. 2124. 2125. 2126. 2127. 2128. 2129. 2130. 2131. 2132. 2133. 2134. 2135. 2136. 2137. 2138. 2139. 2140. 2141. 2142. 2143. 2144. 2145. 2146. 2147. 2148. 2149. 2150. 2151. 2152. 2153. 2154. 2155. 2156. 2157. 2158. 2159. 2160. 2161. 2162. 2163. 2164. 2165. 2166. 2167. 2168. 2169. 2170. 2171. 2172. 2173. 2174. 2175. 2176. 2177. 2178. 2179. 2180. 2181. 2182. 2183. 2184. 2185. 2186. 2187. 2188. 2189. 2190. 2191. 2192. 2193. 2194. 2195. 2196. 2197. 2198. 2199. 2200. 2201. 2202. 2203. 2204. 2205. 2206. 2207. 2208. 2209. 2210. 2211. 2212. 2213. 2214. 2215. 2216. 2217. 2218. 2219. 2220. 2221. 2222. 2223. 2224. 2225. 2226. 2227. 2228. 2229. 2230. 2231. 2232. 2233. 2234. 2235. 2236. 2237. 2238. 2239. 2240. 2241. 2242. 2243. 2244. 2245. 2246. 2247. 2248. 2249. 2250. 2251. 2252. 2253. 2254. 2255. 2256. 2257. 2258. 2259. 2260. 2261. 2262. 2263. 2264. 2265. 2266. 2267. 2268. 2269. 2270. 2271. 2272. 2273. 2274. 2275. 2276. 2277. 2278. 2279. 2280. 2281. 2282. 2283. 2284. 2285. 2286. 2287. 2288. 2289. 2290. 2291. 2292. 2293. 2294. 2295. 2296. 2297. 2298. 2299. 2300. 2301. 2302. 2303. 2304. 2305. 2306. 2307. 2308. 2309. 2310. 2311. 2312. 2313. 2314. 2315. 2316. 2317. 2318. 2319. 2320. 2321. 2322. 2323. 2324. 2325. 2326. 2327. 2328. 2329. 2330. 2331. 2332. 2333. 2334. 2335. 2336. 2337. 2338. 2339. 2340. 2341. 2342. 2343. 2344. 2345. 2346. 2347. 2348. 2349. 2350. 2351. 2352. 2353. 2354. 2355. 2356. 2357. 2358. 2359. 2360. 2361. 2362. 2363. 2364. 2365. 2366. 2367. 2368. 2369. 2370. 2371. 2372. 2373. 2374. 2375. 2376. 2377. 2378. 2379. 2380. 2381. 2382. 2383. 2384. 2385. 2386. 2387. 2388. 2389. 2390. 2391. 2392. 2393. 2394. 2395. 2396. 2397. 2398. 2399. 2400. 2401. 2402. 2403. 2404. 2405. 2406. 2407. 2408. 2409. 2410. 2411. 2412. 2413. 2414. 2415. 2416. 2417. 2418. 2419. 2420. 2421. 2422. 2423. 2424. 2425. 2426. 2427. 2428. 2429. 2430. 2431. 2432. 2433. 2434. 2435. 2436. 2437. 2438. 2439. 2440. 2441. 2442. 2443. 2444. 2445. 2446. 2447. 2448. 2449. 2450. 2451. 2452. 2453. 2454. 2455. 2456. 2457. 2458. 2459. 2460. 2461. 2462. 2463. 2464. 2465. 2466. 2467. 2468. 2469. 2470. 2471. 2472. 2473. 2474. 2475. 2476. 2477. 2478. 2479. 2480. 2481. 2482. 2483. 2484. 2485. 2486. 2487. 2488. 2489. 2490. 2491. 2492. 2493. 2494. 2495. 2496. 2497. 2498. 2499. 2500. 2501. 2502. 2503. 2504. 2505. 2506. 2507. 2508. 2509. 2510. 2511. 2512. 2513. 2514. 2515. 2516. 2517. 2518. 2519. 2520. 2521. 2522. 2523. 2524. 2525. 2526. 2527. 2528. 2529. 2530. 2531. 2532. 2533. 2534. 2535. 2536. 2537. 2538. 2539. 2540. 2541. 2542. 2543. 2544. 2545. 2546. 2547. 2548. 2549. 2550. 2551. 2552. 2553. 2554. 2555. 2556. 2557. 2558. 2559. 2560. 2561. 2562. 2563. 2564. 2565. 2566. 2567. 2568. 2569. 2570. 2571. 2572. 2573. 2574. 2575. 2576. 2577. 2578. 2579. 2580. 2581. 2582. 2583. 2584. 2585. 2586. 2587. 2588. 2589. 2590. 2591. 2592. 2593. 2594. 2595. 2596. 2597. 2598. 2599. 2600. 2601. 2602. 2603. 2604. 2605. 2606. 2607. 2608. 2609. 2610. 2611. 2612. 2613. 2614. 2615. 2616. 2617. 2618. 2619. 2620. 2621. 2622. 2623. 2624. 2625. 2626. 2627. 2628. 2629. 2630. 2631. 2632. 2633. 2634. 2635. 2636. 2637. 2638. 2639. 2640. 2641. 2642. 2643. 2644. 2645. 2646. 2647. 2648. 2649. 2650. 2651. 2652. 2653. 2654. 2655. 2656. 2657. 2658. 2659. 2660. 2661. 2662. 2663. 2664. 2665. 2666. 2667. 2668. 2669. 2670. 2671. 2672. 2673. 2674. 2675. 2676. 2677. 2678.

O CARTÃO-ÍNDICE ABREVIADO ocupa 128 páginas, possuindo, na 1ª edição, Principais, contendo as 448 páginas sempre que uma chave de parâmetros estiver disponível. Assim sendo, nenhuma operação de busca é realizada em posições a partir do endereço (1801)<sub>16</sub>.

A linguagem de CARICATONE ABSOLUTO é compatível com o Sistema Principal de câlculos de um programa em formato ABSOLUTO montado em uma fita de papel perfurada, utilizando a CÍTTICA DE FITAS PERFORADAS.

$$H_1 = H_0 \cup \{p_1, \dots, p_n\} \quad \text{with } p_1, \dots, p_n \in H_0 \text{ and } p_i \neq p_j \text{ for } i \neq j.$$

1. Estado de espera

O "Teste de zero" (TESTE ZERO) representa a busca por zeros entre as palavras de palavras produzidas no processo de execução.

No teste de zero, a mesma operação de busca é realizada em todos os computadores, com a diferença de que a busca de verificação da existência de zeros de leitura.

No caso de erro de leitura, o CARRICADOR ABSOLUTO, através do MOSTRADOR DO ACUMULADOR de pontos de leitura, indica o ponto de erro.

Se o transporte do programa for realizado com o MOSTRADOR DO ACUMULADOR = 00, os valores  $P_0, P_1, P_2$  são automaticamente postos 0 e 1 da Memória e é executada uma instrução de PASE.

Após o teste de leitura PASTIDA, o CARRICADOR ABSOLUTO executa um dos seguintes procedimentos para a posição 0 executando portanto a execução do programa que foi armazenado.

A 128 posições condicionadas pelo CARRICADOR ABSOLUTO, em 128 posições da Memória, é transportado por meio de um processo chamado "PRI-CARRICADOR" (128 posições) para o PASE. A posição de 128 posições da Memória é posta em 128 posições do CARRICADOR ABSOLUTO por longo período de tempo, até o ponto de teste de Memória de todos os pontos de leitura. Por se tratando de Memória de todos os pontos de leitura, pode-se utilizar-se no mesmo computador, com o mesmo sistema de leitura, os dados de operação de computador e memória. Portanto, os dados de operação de computador e memória, por serem produzidos por meio de um dispositivo de computador, são produzidos por meio de um dispositivo de computador, sendo assim, a utilização de um mesmo computador, em longos períodos, não é necessária.

Devido ao fato de a utilização de um computador, operando em 128 posições, com o transporte de dados, por meio de um dispositivo de computador, não ser necessária, a utilização de um mesmo computador, em longos períodos, não é necessária.

O processo SM 7, executado somente após o fechamento do Sistema Básico de Controle, na Memória Principal por meio do CARICADOR ALCOLQ11,

tem por finalidade iniciar a operação de interface de interrupção síncrona (interfases 1) e de relógio de tempo (interfase 2), criar os procedimentos, com o CPE e através do processo PDI.

A estrutura dos dados do SM7 é armazenada com os vetores de interrupção convenientes, de tal maneira que por meio da execução do SFC, o SM7 adquira a existência própria do processo CPE.

Após a ativação do processo CPE, o processo SM 7 entra no estado PARADO, não sendo mais reativado. A figura 4.1.1 apresenta um diagrama do processo CPE.

#### 4.1.1 - INICIAÇÃO DA INTERFACE 1

A iniciação da interface 1 consiste em enviar o sinal de início da interface com o nível lógico  $(PI)_{1 \rightarrow 0}$  e o nível lógico  $(PI)_{1 \rightarrow 1}$  e ainda preparar o nível de tempo de espera para a primeira interrupção, iniciada que o nível lógico  $(PI)_{1 \rightarrow 0}$  é 0.

#### 4.1.2 - INICIAÇÃO DO RELÓGIO DE TEMPO PDI

O relógio de tempo PDI é iniciado com um período de 100 ms, correspondente a 2 unidades de tempo (U.T.). Este tempo é um período de interrupção e após liberado, com a ocorrência de uma interrupção, que indica o processo que se encontra parado.







TUL

DISP. V	DISP. P

TM


DISP. V

PROCESS

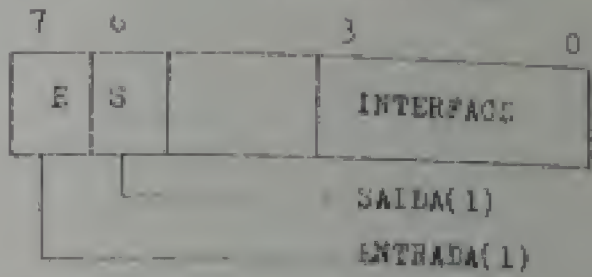
TED

S	S	INTERFACE



A tabela TPI contém a base de dados de correspondência entre o número de dispositivos, no tipo INST.VI e o endereço de tabela (INST.VI) contendo a tabela de dados. A tabela TPI é utilizada para a obtenção de dados de correspondência entre o endereço de tabela e o endereço de dispositivo. A tabela TPI é utilizada para a obtenção de dados de correspondência entre o endereço de tabela e o endereço de dispositivo.

Os correspondentes de cada dispositivo (INST.VI) a cada endereço de tabela, são obtidos através da tabela de correspondência de dados de entrada, saída ou ambas. Quando a tabela de correspondência de dados de entrada, saída ou ambas, estiver conectada, a tabela de correspondência de dados de entrada, saída ou ambas, estará conectada.



Por fim, a tabela TPI contém as correspondências entre o dispositivo (INST.VI) e o endereço de tabela que é utilizado. Quando não estiver sendo utilizado, este elemento de TPI conterá o endereço (00)16.

### 3.1.1 - PARÂMETRO DE CONFIGURAÇÃO DO TABELADO TPI

Os tipos de operação a serem efetuados nos dispositivos de Entrada/Saída são designados mediante o envio de uma mensagem ao processo PLS. Esta mensagem tem as seguintes opções, com a operação a ser exercida por PLS:

#### 1) OPERAÇÃO DE MONTAGEM LÓGICA

SEMPRE (1) 0 1 0 1 0 1

onde as quatro palavras que compõem esta mensagem são interpretadas por:

1 - Endereço de processo remoto  
0 - Endereço local

$x_y$  - Unidade lógica

$y_y$  - Unidade lógica

31.

Este comando faz com que o dispositivo (DISP.F) de saída  
do  $y$  UNIDADE LÓGICA  $y_y$  passe a ser o dispositivo (DISP.F) de saída  
do  $x$  UNIDADE LÓGICA  $x_y$ .

Na versão inicial do Sistema Básico de Controle, a correspondência entre DISP.F e os dispositivos periféricos utilizados é a seguinte:

DISP.F PERIFÉRICO

- 0 Teletipografia (TTY)
- 1 Teletipografia (TTY)
- 2 Leitora de fita de papel
- 3 Perfuradora rápida de fita de papel
- 4 Impressora de fita
- 5 Leitora de cartões

#### 6) OPERAÇÃO DE ENTRADA/SÁIDA

ENTRADA  $x$   $x_1$   $x_2$   $x_3$   $x_4$   $x_5$   $x_6$   $x_7$   $x_8$   $x_9$   $x_{10}$   $x_{11}$   $x_{12}$   $x_{13}$   $x_{14}$   $x_{15}$

Neste caso,  $x$  será dada a função de processar os dados  
de  $x_1$  a  $x_{15}$  e a saída será dada a partir de  $x_{16}$  a  $x_{31}$ .  
A função de processar os dados de  $x_1$  a  $x_{15}$  será dada a  
função de processar os dados de  $x_{16}$  a  $x_{31}$ .  
A função de processar os dados de  $x_{16}$  a  $x_{31}$  será dada a  
função de processar os dados de  $x_{16}$  a  $x_{31}$ .

A função de processar os dados de  $x_{16}$  a  $x_{31}$  será dada a  
função de processar os dados de  $x_{16}$  a  $x_{31}$ .

7	6	5	4	3	2	1	0
0	1	2	3	4	5	6	7

- 0/5 - Entrada (0) ou saída (1)
- 0/6 - Por caracteres (0) ou binária (1)
- 0/7 - Entrada zero (0) ou saída (1)
- 0/8 - Unidade lógica

A unidade de análise de dados por processo é dada por (2.3.1) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ . A unidade de análise de dados por indivíduo é dada por (2.3.2) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ . A unidade de análise de dados por processo e indivíduo é dada por (2.3.3) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ .

A unidade de análise de dados por processo e indivíduo é dada por (2.3.4) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ .

A unidade de análise de dados por processo e indivíduo é dada por (2.3.5) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ .

A unidade de análise de dados por processo e indivíduo é dada por (2.3.6) sendo as variáveis dependentes em  $Y$  e as independentes em  $X$ .

Processo	1	2	3	4	5	6	7	8
1	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0
4	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0
6	0	0	0	0	0	1	0	0
7	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	1

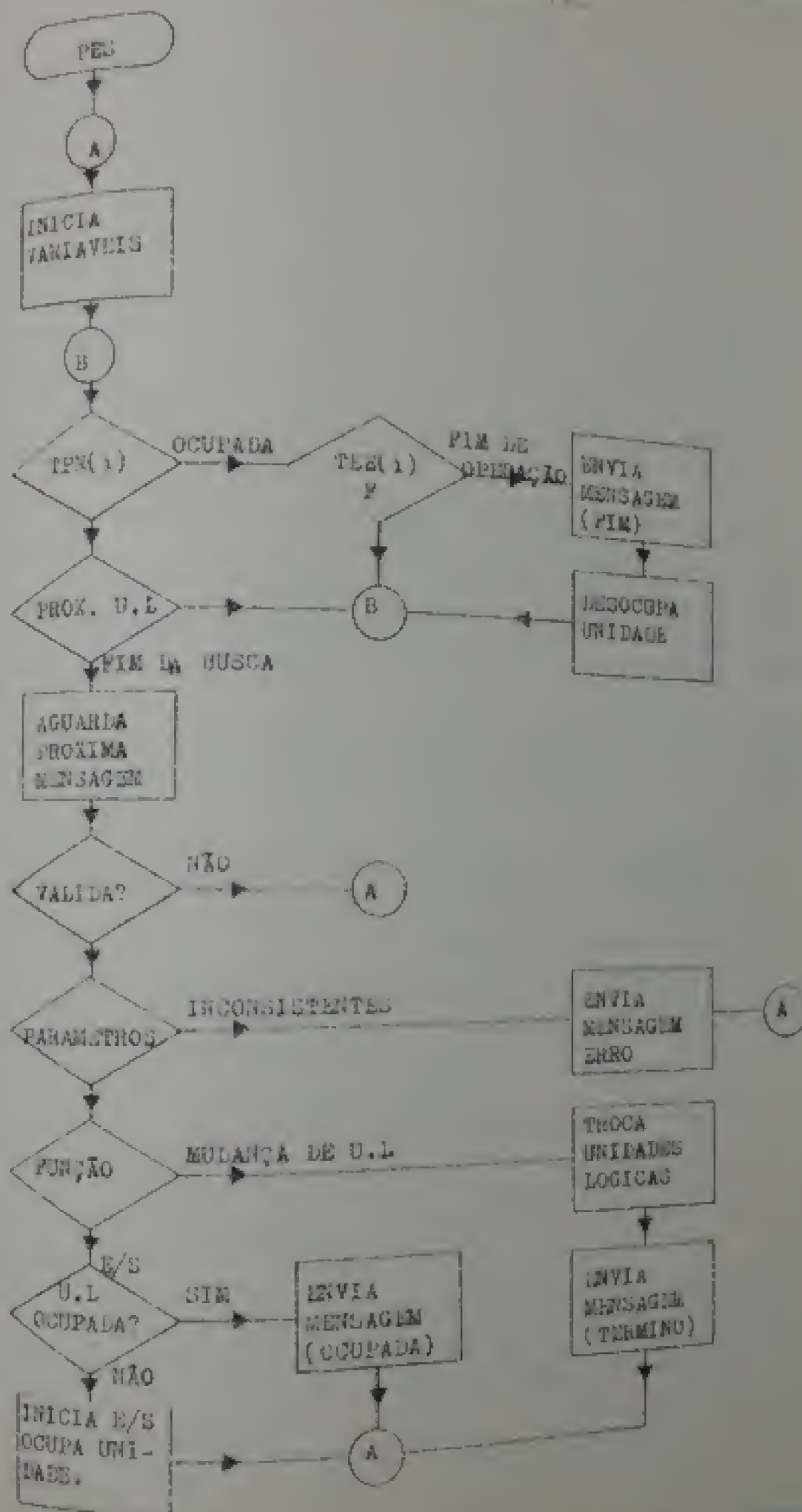
Após a escolha de uma unidade de análise de dados, o processo de análise de dados é iniciado. O primeiro passo é a escolha da unidade de análise de dados. O segundo passo é a escolha da unidade de análise de dados. O terceiro passo é a escolha da unidade de análise de dados.

Após a escolha de uma unidade de análise de dados, o processo de análise de dados é iniciado. O primeiro passo é a escolha da unidade de análise de dados. O segundo passo é a escolha da unidade de análise de dados. O terceiro passo é a escolha da unidade de análise de dados.

### 3.3.3 - ESTRUTURA DO PROCESSO PES

O processo PES é estruturado de acordo com a seguinte forma: a) escolha da unidade de análise de dados; b) escolha da unidade de análise de dados; c) escolha da unidade de análise de dados; d) escolha da unidade de análise de dados; e) escolha da unidade de análise de dados.





Para a operação de Mediana de Unidade Técnica não pode ser realizada se o processo resultante não for autorizado, dependendo das operações de Intermédiação. O não é iniciada se a Unidade Técnica pedida a favor também utilizada por outro processo. Se estes casos uma situação semelhante é retornada ao processo resultante.

# 3.4.1 - PARÂMETROS DE COMPARAÇÃO ENTRE PRÓCESSOS

3.

A ordem do processo (PR) para uma certa realização  $\omega$  de um sistema  $\tilde{S}$  realizada mediante o envio de uma mensagem de solicitação dos parâmetros, dependerá do histórico observado.

## 3.4.1.1 - PRÓCESSO ASSÍNCRONO

PARÂMETROS:  $K$ ,  $F$ ,  $P.L.$ ,  $C_1$ ,  $P_m$

onde  $K$  = código do processo observado.

$F$  = formato igual a 0 (zero).

$P.L.$  = unidade básica de origem de arquivo.

$C_1, P_m$  = endereço no Memória Principal para a busca de parâmetros.

O endereço  $C_1, P_m$  é contido em aquele existente no mesmo endereço (3.4.1.1) disponível no estado inicial referido e por ser diferente, o pedido é repetido.

## 3.4.1.2 - PRÓCESSO SÍNCRONO

PARÂMETROS:  $K$ ,  $F$ ,  $P.L.$ ,  $R_1$ ,  $R_m$

$K$  = código do processo observado.

$F$  = formato igual a 1 (um).

$P.L.$  = unidade básica de origem de arquivo.

$R_1, R_m$  = base de relocação.

Em qualquer do processo no formato relativo é considerado por este tipo de bloco, cada um com no máximo 17 palavras.

O tipo de formatização que tem a palavra  $m$  no endereço  $i$  (onde  $i$  é um número inteiro de 0 a 16) de base  $B$  (3.4.1.2).

Os tipos de formatização são:



$q$  = número de palavras do dicionário considerado no teste  
 $k_1, k_2, k_3$  = graus de liberdade para o teste de hipótese  
 $\alpha$  = nível de significância  
 $df_1, df_2$  = graus de liberdade de cada uma das variáveis  
 $T$  = teste de hipótese

$\mu_1, \mu_2$  = médias das variáveis

$\sigma_1, \sigma_2$  = desvios padrão das variáveis

$\pi_1, \pi_2$  = probabilidades de ocorrência de cada uma das variáveis

$\rho_1, \rho_2$  = coeficientes de correlação

$r$  = teste de hipótese

Cada elemento do grupo de teste  $T = \{x_1, x_2, \dots, x_n\}$  é considerado um elemento de localização  $x$  de cada uma das variáveis  $x_1, x_2, \dots, x_n$ . O teste de hipótese é realizado considerando o teste de hipótese de localização  $x$  de cada uma das variáveis  $x_1, x_2, \dots, x_n$ .

Grupo de teste de hipótese:  $T_1, T_2, T_3, T_4, T_5$

$T_1$  = teste de hipótese absoluta (uma palavra)  
 $T_2$  = teste de hipótese relativa (uma palavra)  
 $T_3$  = teste de hipótese relativa (duas palavras)  
 $T_4$  = teste de hipótese relativa (três palavras)  
 $T_5$  = teste de hipótese relativa (quatro palavras)

$(0, 1, 2, 3)$

O teste de hipótese é realizado considerando o teste de hipótese de localização  $x$  de cada uma das variáveis  $x_1, x_2, \dots, x_n$ . O teste de hipótese é realizado considerando o teste de hipótese de localização  $x$  de cada uma das variáveis  $x_1, x_2, \dots, x_n$ .

$\mu_1, \mu_2$  = médias das variáveis  
 $\sigma_1, \sigma_2$  = desvios padrão das variáveis  
 $\pi_1, \pi_2$  = probabilidades de ocorrência de cada uma das variáveis  
 $\rho_1, \rho_2$  = coeficientes de correlação

$r$  = teste de hipótese

S	12	11	10	C	D11	D20	T
---	----	----	----	---	-----	-----	---

BLOCO DE IDENTIFICAÇÃO

BLOCO DE DADOS

S	BEL	EO	RO	D00	D10	D20	D30	...	Rj	D0j	D1j	D2j	D3j	...
---	-----	----	----	-----	-----	-----	-----	-----	----	-----	-----	-----	-----	-----

...	Rn	D0n	D1n	D2n	D3n	T
-----	----	-----	-----	-----	-----	---

n=NE

Fig. - 1.2

ORGANIZAÇÃO DO CANAL DE TRANSMISSÃO

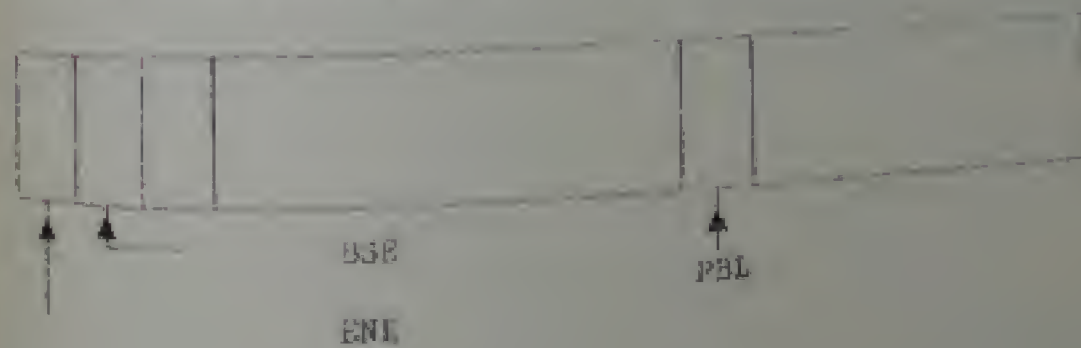


Fig. - 1.3

O processo de ajuste de longo prazo das variáveis reais é estudado no capítulo 3.3.2. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

3.3.3 - INTERPRETAÇÃO DO MODELO DE PPP

O processo de ajuste de longo prazo das variáveis reais é estudado no capítulo 3.3.2. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

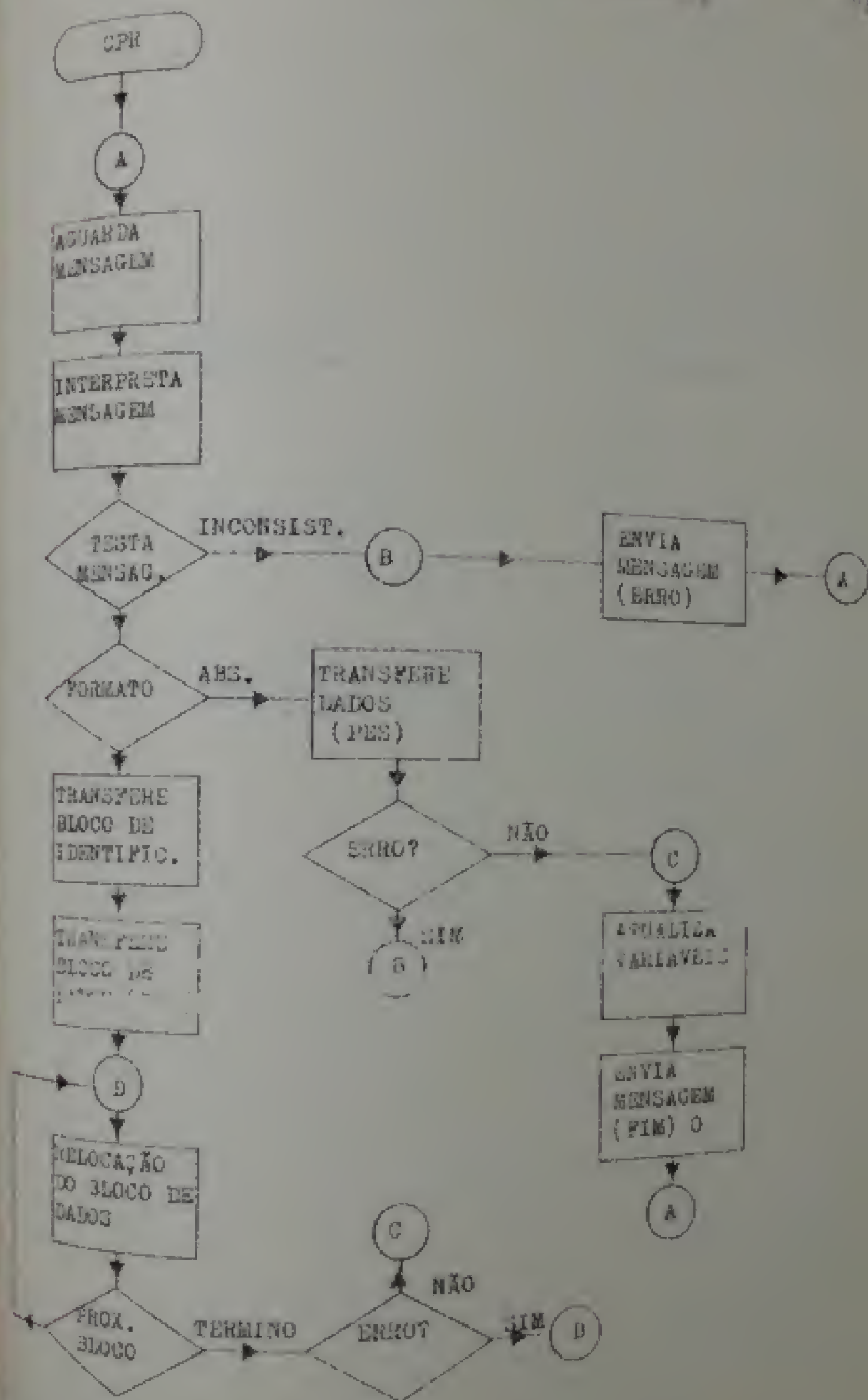
A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

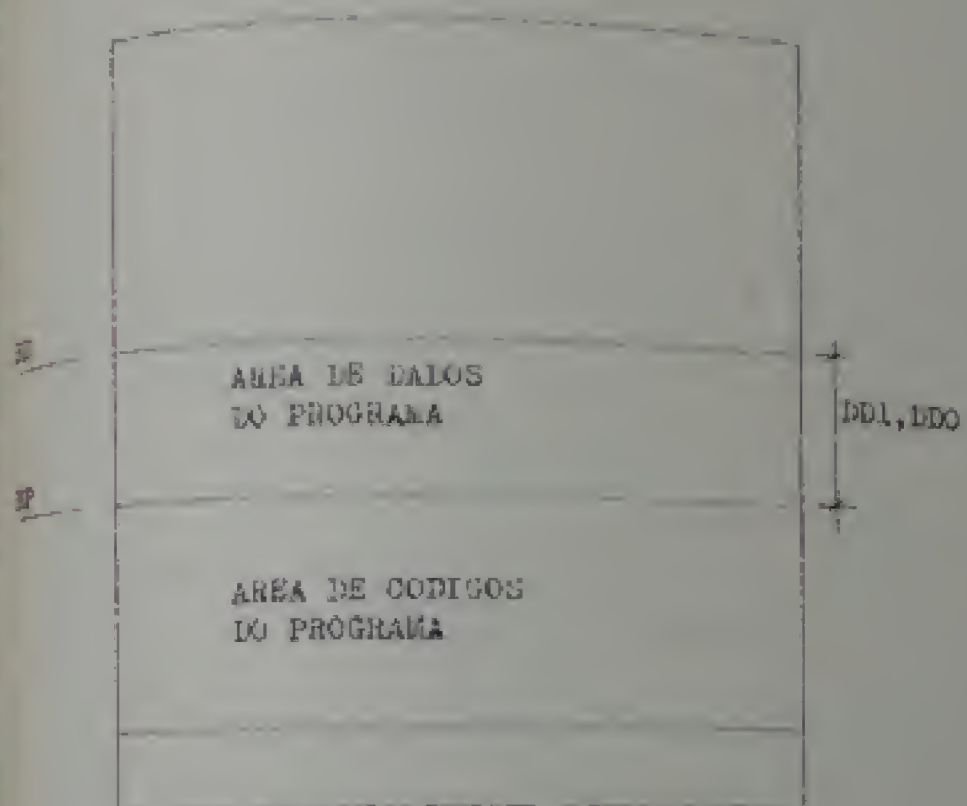
A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.

A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3. A interpretação do modelo de PPP é dada no capítulo 3.3.3.







MEMORIA PRINCIPAL

O procedimento de cada bloco de dados é feito a partir dos dados anteriores e calculado como:  $(x_{i+1} + 1) \cdot (1 + x_{i+1})$  ou  $(x_{i+1} + 1) \cdot (1 + x_{i+1})$  no início de cada bloco  $(1, 4, 10)$ .

Calculado o endereço de armazenamento, os dados são armazenados consecutivamente a partir deste endereço  $E$ .

Após os armazenamento, cada dado sofre uma modificação  $(MOD(X, Y))$  de acordo com o código de alteração correspondente. Assim, o dado  $x$  por exemplo muda para  $(x + 1)$  ou  $(x - 1)$  por:

$$MOD(x, 1) = x + 1 \quad \text{ou} \quad MOD(x, 2) = x - 1$$

$$MOD(x, 3) = x + 10 \quad \text{ou} \quad MOD(x, 4) = x - 10$$

$$MOD(x, 5) = x + 100 \quad \text{ou} \quad MOD(x, 6) = x - 100$$

com  $x = 1, 2, 3, \dots, 1000$  e  $MOD(x, 7) = MOD(x, 8) = MOD(x, 9) = MOD(x, 10) = 0$ .

A seleção ocorre de acordo com o mesmo programa existente de acesso ao arquivo, ou seja, para cada índice é executado um programa para ler o endereço principal e a modificação de cada modificação. Isto ocorre necessariamente quando se trata de um arquivo no formato sequencial, ou seja, de acesso de ENDEREÇO DE ACÉSSO SEQUENCIAL ou seja, de acesso de "FATIAMENTO SEQUENCIAL".

Finalmente, a distribuição de programas é realizada em 3 partes: primeira parte de 1 a 10, segunda parte de 11 a 100, e terceira parte de 101 a 1000.

#### 2.3.3 - DISTRIBUIÇÃO DOS DADOS DE PROGRAMA

O processo de distribuição dos dados de programa é realizado de acordo com o mesmo programa de distribuição de dados de programa. A distribuição consiste em distribuir os dados de programa de acordo com o código de alteração correspondente. Assim, o dado  $x$  por exemplo muda para  $(x + 1)$  ou  $(x - 1)$  por:

O programa  $(1, 4, 10)$  de acordo com o código de alteração correspondente. Assim, o dado  $x$  por exemplo muda para  $(x + 1)$  ou  $(x - 1)$  por:





01 01E31 2.24' 50

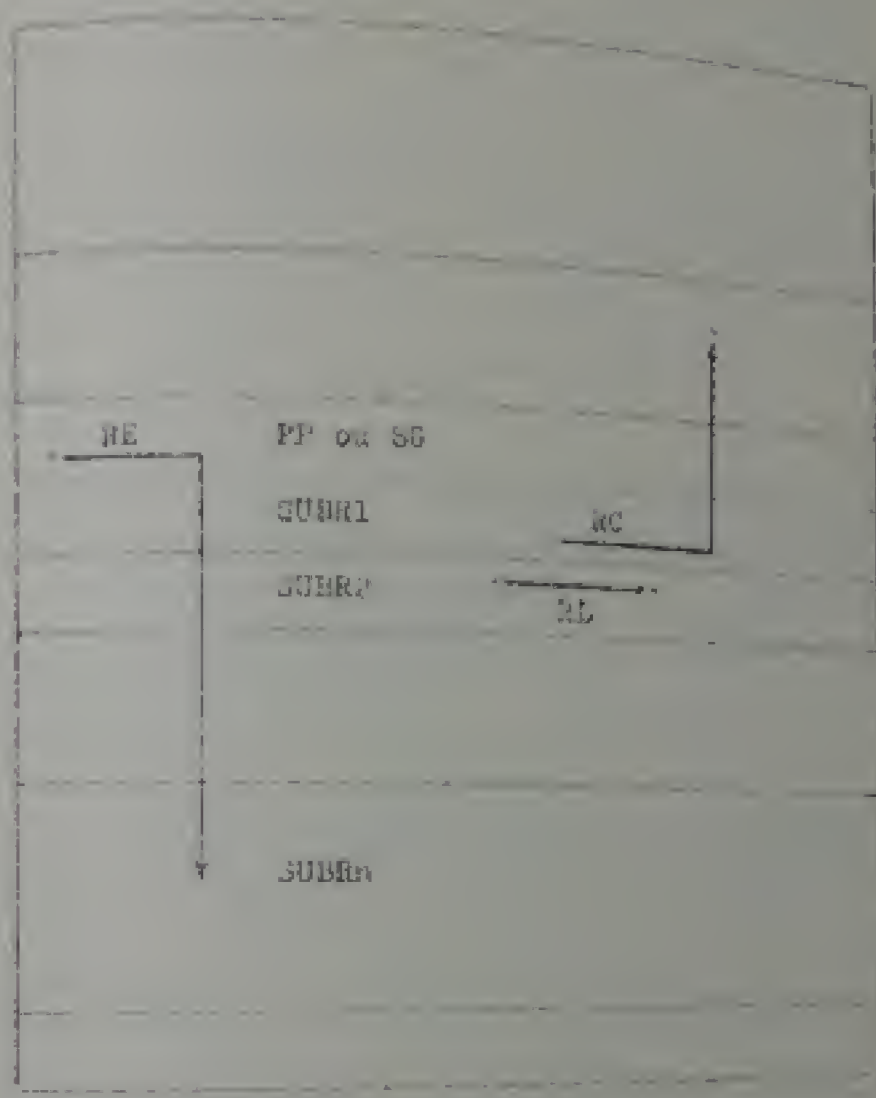
BR

OFF

OS1

OS2

OSn



AREA  
COMUN  
DO LALOS

ME0

ME1

ME2

MEn



17

...do ... ..  
... ..  
... ..  
... ..

... ..  
... ..  
... ..  
... ..

... ..

... ..  
... ..  
... ..  
... ..  
... ..  
... ..

... ..  
... ..  
... ..  
... ..  
... ..  
... ..

... ..  
... ..  
... ..  
... ..

... ..  
... ..

... ..  
... ..  
... ..  
... ..  
... ..

uma vez que utilizamos o sistema KILBY, o primeiro  
quando armazenamos em memória uma nova relação. O L. 11, 12 e  
13 são os CARACTERES REFORÇOS. O L. 14 é o  
L. 15 é a D. 16 é a D. 17 é a D. 18 é a D. 19 é a D. 20 é a D.

#### 1.5 - DESCRIÇÃO DO PROCESSO DE COMUNICAÇÃO COM

O processo de comunicação COM estabelece uma interação  
entre o operador, o sistema e outros processos em andamento  
no Sistema Básico de Controle.

Esta interação se processa por meio de diretivas e  
valores enviados através de uma teleimpressora (ver anexo 11).

As diretivas ou comandos enviados pelo operador são  
interpretadas e executadas pelo processo COM, após o qual este  
envia uma mensagem ao operador. Na execução destas diretivas pode  
haver a intervenção de outros processos, o que é feito pelo processo  
COM, tendo interação com outros processos permite ao operador  
verificar um controle sobre o funcionamento de todo o Sistema Básico  
de Controle.

#### 1.5.1 - DIRETIVAS ENVIADAS DO PROCESSO COM

Uma diretiva é constituída por um conjunto de no máximo  
40 caracteres ASCII enviados ao processo COM por meio de uma  
teleimpressora. De acordo com a interpretação e o resultado da  
execução da diretiva, uma ou mais mensagens são enviadas ao  
operador através da mesma teleimpressora.

As especificações dos comandos operacionais são  
definidas no processo COM e são enviadas ao operador através  
da tabela de sig. 1.11.

De acordo com a sig. 1.11, cada diretiva é constituída por  
um conjunto de caracteres ASCII definidos por uma série de dígitos  
e caracteres de controle. Os caracteres de controle são  
definidos em tabelas de sig. 1.11 e 1.12. Os caracteres de  
dados são definidos em tabelas de sig. 1.13 e 1.14.

1. Cada derivado superior a segundo da função  $f(x)$ , para  $x$  em  $(a, b)$ , satisfaz a equação diferencial  $y'' + p(x)y' + q(x)y = r(x)$ , onde  $p, q, r$  são funções contínuas em  $[a, b]$ .

2. Se  $y_1, y_2, \dots, y_n$  são soluções da equação homogênea  $y'' + p(x)y' + q(x)y = 0$ , então qualquer combinação linear  $y = c_1 y_1 + c_2 y_2 + \dots + c_n y_n$  também é solução da equação homogênea.

3. Se  $y_1, y_2, \dots, y_n$  são soluções da equação homogênea e  $y_p$  é uma solução particular da equação não homogênea  $y'' + p(x)y' + q(x)y = r(x)$ , então  $y = y_1 c_1 + y_2 c_2 + \dots + y_n c_n + y_p$  é a solução geral da equação não homogênea.

4. A equação diferencial  $y'' + p(x)y' + q(x)y = r(x)$  pode ser resolvida por métodos de integração direta, separação de variáveis, ou métodos de redução de ordem, dependendo da forma das funções  $p, q, r$ .





uma vez conhecida a derivada, encontra-se a integral -  
 a qual é a função primitiva da derivada. A integral é  
 determinada até uma constante, pois a derivada de uma  
 constante é zero. A integral de uma função é a função  
 primitiva da mesma.

1.1.1.1

Exemplo 1: integral da derivada de  $f(x)$ .

Tem-se a integral da derivada de  $f(x)$  em relação a  $x$ .

1.1

De acordo com a definição de integral, a integral de uma  
 função é a função primitiva da mesma.

1.1.1.2, ..., 1.1.1.10

Este exemplo corresponde à integral de uma função em  
 relação a  $x$ , por um outro processo, ou seja, a integral  
 de um primitivo  $f(x)$ ,  $f'(x)$ , em  $x$ , isto é, a integral de  
 uma derivada em relação a  $x$  é a função primitiva da  
 mesma.

A integral de uma função em relação a  $x$  é a função  
 primitiva da mesma, ou seja, a integral de uma derivada  
 em relação a  $x$  é a função primitiva da mesma.

1.1.1.11, ..., 1.1.1.12

Integral dos valores de uma função em relação a  $x$ .

1.1.1.13 - INTEGRAL DE VALORES EM FUNÇÃO DE  $x$

A integral de uma função em relação a  $x$  é a função  
 primitiva da mesma, ou seja, a integral de uma derivada  
 em relação a  $x$  é a função primitiva da mesma.



TABELA DE INTERPRETAÇÃO DAS DIRETIVAS (TID)

T1A	T1B	T1C
D1	D2	L3



Para a interpretação das diretrizes, a primeira coisa a fazer é ler a Tabela de Interpretação das Diretrizes (TID) contida no Anexo 1.1.1.1. A TID contém as diretrizes e as suas respectivas explicações. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.

### 3.1.1.1 - TABELA DE INTERPRETAÇÃO

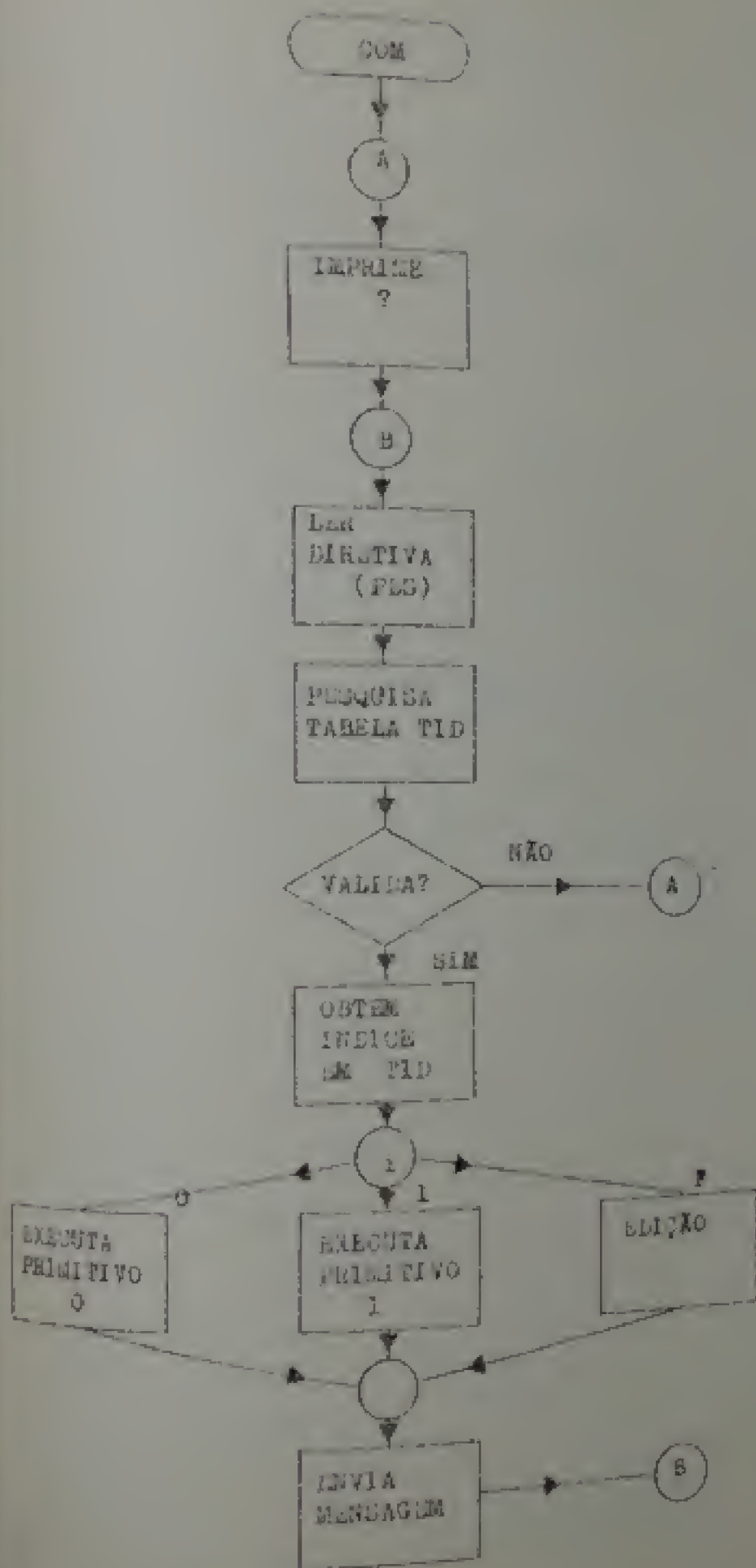
Inicialmente o caráter é impresso, mostrando o ponto de partida de uma diretiva, por meio do primeiro número da TID. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.

De acordo com a TID, a primeira coisa a ser lida antes de se ler as diretrizes é a TID. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.

Logo, a primeira coisa a ser lida antes de se ler as diretrizes é a TID. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.

Indicação de como interpretar a TID. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.

Indicação de como interpretar a TID. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes. A TID é a primeira coisa a ser lida antes de se ler as diretrizes.



CAPITULO IV. DE LA VIDA DE LOS REYES.





13. 1991.10.20 (日) 晴 19.5℃ 19.1℃ 15.5℃

100

deste modo, um novo tratamento deve ser dado ao tratamento  
dos dados, não sendo suficiente simplesmente aplicar o mesmo  
tratamento, podendo eventualmente fazer-se ajustes.

#### EXERCÍCIOS DO PRÁTICO

A descrição dos processos, aliada, com a execução  
de um programa destinado para cada situação, com a execução  
e a interpretação dos resultados, que se constitui de um processo, com  
uma avaliação do processo, constituem a base.

Este livro não pretende ser mais do que um guia para se  
aproximar de um único processo, característico de um modo  
util em situações de tempo compartilhado ("Time sharing") (p. 11).

Uma modificação nesta descrição que possa ser feita  
depois da caracterização consiste em colocar os elementos que  
formam o "TAP" de um processo em colunas adjacentes, de modo  
a facilitar a leitura dos dados (TAP), mencionado no capítulo 2, no  
lugar das informações de descrição. De modo, a cada processo  
tem associado um sistema independente de processo que se aplica  
nesta.

Para isso, além da necessidade de um tipo de acesso de  
modo para as colunas adjacentes de cada TAP, a leitura de cada  
coluna de processos (TAP) e a leitura de cada sistema de processo  
deve ser feita.

#### EXERCÍCIOS DO PRÁTICO

A organização da interface de um sistema de  
processos, aliada com a execução de um programa, com a execução  
e a interpretação dos resultados, que se constitui de um processo, com  
uma avaliação do processo, constituem a base.





181

...no sistema, para garantir que os recursos sejam utilizados de forma adequada e para evitar a duplicação de esforços. O sistema deve ser capaz de identificar e corrigir erros de digitação e de garantir a integridade dos dados. Além disso, o sistema deve ser capaz de gerar relatórios e estatísticas que possam ser utilizados para a tomada de decisões.

...a partir da proteção contra violações de dados. O sistema deve ser capaz de identificar e corrigir erros de digitação e de garantir a integridade dos dados. Além disso, o sistema deve ser capaz de gerar relatórios e estatísticas que possam ser utilizados para a tomada de decisões.

...a partir da proteção contra violações de dados. O sistema deve ser capaz de identificar e corrigir erros de digitação e de garantir a integridade dos dados. Além disso, o sistema deve ser capaz de gerar relatórios e estatísticas que possam ser utilizados para a tomada de decisões.

#### 4.1.4 - CRIAÇÃO DE NOVA

A possibilidade de expansão de dados é o principal fator que permite a utilização de dados em tempo real e a criação de novos dados.

O sistema de expansão proposto no capítulo 4.1.4 pode ser utilizado inclusive como sistema de backup de dados.

A inclusão de novos dados deve ser feita de forma segura e controlada, evitando a duplicação de dados e a perda de dados.

#### 4.1.5 - CRIAÇÃO DE NOVA

Uma das finalidades da implementação do sistema é a criação de novos dados e a manutenção dos dados existentes.

...a partir da proteção contra violações de dados. O sistema deve ser capaz de identificar e corrigir erros de digitação e de garantir a integridade dos dados. Além disso, o sistema deve ser capaz de gerar relatórios e estatísticas que possam ser utilizados para a tomada de decisões.

170.  
de preterito do verbo "ser", em tempo do pretérito perfeito do verbo "fazer", e assim, a expressão "fazer" é substituída por "ser", e a expressão "deprender" é substituída por "fazer".

Logo a simplicidade da expressão "fazer" é substituída por "deprender", e a expressão "deprender" é substituída por "fazer".

Logo a simplicidade da expressão "fazer" é substituída por "deprender", e a expressão "deprender" é substituída por "fazer".

Logo a simplicidade da expressão "fazer" é substituída por "deprender", e a expressão "deprender" é substituída por "fazer".

Logo a simplicidade da expressão "fazer" é substituída por "deprender", e a expressão "deprender" é substituída por "fazer".



APÊNDICE 1

Neste apêndice são apresentadas as principais características das rotinas mais importantes utilizadas na implementação do sistema básico.

Para cada rotina são apresentadas a sua função, variáveis utilizadas, parâmetros de entrada e saída e uma breve descrição de seu funcionamento.

a) CÓDIGO: RDI

b) FUNÇÃO:

Teste dos pedidos de interrupção das interfaces e implementação do esquema de prioridades.

c) CHAMADA:

Acessível por interrupção

d) PARÂMETROS DE ENTRADA:

Nenhum

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS

Nenhuma

h) ROTINAS UTILIZADAS:

ISS, ISR, DCM

i) DESCRIÇÃO:

Realiza uma sequência de testes dos pedidos de interrupção das interfaces e desvios incondicionais para as rotinas responsáveis pelo tratamento de interrupção detectada.



a) CÓDIGO: ISR

b) FUNÇÃO: Tratamento das interrupções geradas pelo Relógio de Tempo Real (R.T.R.)

c) CHAMADA:

acessível por interrupção do R.T.R.

d) PARÂMETROS DE ENTRADA

Nenhum

e) PARÂMETROS DE SAÍDA

Nenhum

f) VARIÁVEIS LOCAIS

RIT, RIA, RIR - Contador de interrupções

g) VARIÁVEIS GLOBAIS

NDT

h) ROTINAS UTILIZADAS

RST, RST, CMP, PAT

i) DESCRIÇÃO

A cada interrupção do R.T.R. a variável RIT, RIA, RIR é incrementada. Um novo processo é seleccionado por meio da rotina PAT e, através da rotina CMP, este processo é colocado em execução.

1 - TITULO DE RESOLUÇÃO DO PRÓXIMO PROCESSO ATIVO

114

a) CÓDIGO: PAT

b) FUNÇÃO:

Busca o próximo processo no estado ATIVO

c) CHAMADA:

PGG PAT

d) PARÂMETROS DE ENTRADA

PEX = Código do processo em execução

e) PARÂMETROS DE SAÍDA

ACUMULADOR: Código do próximo processo ATIVO

f) VARIÁVEIS LOCAIS

NENHUMA

g) VARIÁVEIS GLOBAIS

PEX, TPG

h) FÓRMULAS UTILIZADAS

PAL

i) DESCRIÇÃO:

A partir do número contido em PEX, a Tabela de Referência dos processos (TPG) é percorrida. O código do primeiro processo encontrado no estado ATIVO será carregado no ACUMULADOR.

a) CÓDIGO: ATP

b) FUNÇÃO:

Faz com que um processo repita a emissão de um primitivo

c) CHAMADA:

PUG ATP

d) PARÂMETROS DE ENTRADA:

Nenhum

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

NDT

h) ROTINAS UTILIZADAS:

PAT

i) DESCRIÇÃO:

A variável NDT indicadora do número de parâmetros do primitivo é feita igual a -2. Com isto o CONTADOR DE INSTRUÇÕES do processo passará a apontar a instrução correspondente ao último primitivo emitido pelo processo. Um novo processo é acionado por ação de PAT.



a) CÓDIGO: 155

b) FUNÇÃO:

Tratamento das interrupções síncronas.

c) CHAMADA:

Acessível por interrupção

d) PARÂMETROS DE ENTRADA:

Endereço de interrupção (posições 2 e 3 da memória)

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

LPI, EPI - conteúdo das posições de interrupção (2 e 3)  
 NBT - Número de parâmetros transitíveis  
 NPP - Número de primitivas  
 TDS - Tabela de Estados dos processos  
 TDI - Tabela dos Descritores dos processos  
 TDC - Tabela dos Descritores dos processos  
 TDH - Tabela de Códigos dos processos

g) VARIÁVEIS GLOBAIS:

Nenhuma

h) ROTINAS UTILIZADAS:

SST, TPI, RST, PRT

i) DESCRIÇÃO:

O "STATUS" do processo que provocou a interrupção é salvo por meio da rotina SST. A variável LPI, EPI passa a apontar os parâmetros relacionados com o primitivo a ser executado. Por meio da rotina TPI e da variável NBT estes são transferidos. O primeiro

os parâmetros é verificada comparando-se com o valor de  $\Delta P$ , após  
e que uma subrotina de PGM é escolhida de acordo com este parâmetro.

No retorno de  $\Delta P$ , o conteúdo da posição 2 e 3 da memória  
é incrementado de valor de  $\Delta P$  e 128 de atualizar a endereço de  
retorno da interrupção, a perda de interrupção da interface. O  
contêiner de interrupção de  $\Delta P$  é lido igual a zero, o "STATUS" do programa interrompido é  
recuperado por meio da rotina 257 e finalmente é executada a rotina 258.

a) CÓDIGO: NST

b) FUNÇÃO:

Salvar provisoriamente o "STATUS" de um processo interrompido.

c) CHAMADA:

PGC NST

d) PARÂMETROS DE ENTRADA:

Nenhum

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

STA, STN, STI, STT

g) VARIÁVEIS GLOBAIS:

Nenhuma

h) ROTINAS UTILIZADAS:

Nenhuma

i) DESCRIÇÃO:

Os valores do ACUMULADOR, EXTENSÃO e ÍNDICE são armazenados nas variáveis STA, STI e STI respectivamente. Os bits de TRASELOR e "VAL-INT" são colocados nos bits 7 e 6 respectivamente da variável STT.



a) CÓDIGO: RST

b) FUNÇÃO:

Restaurar o "STATUS" de um processo interrompido e que deve continuar em execução.

c) OPERANDA:

RPC RST

d) PARÂMETROS DE ENTRADA:

Nenhum

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

STA, STI, STJ, STT

h) ROTINAS UTILIZADAS:

Nenhuma

i) DESENCIAÇÃO:

O ACUMULADOR, EXTENSÃO e ÍNDICE são carregados com os valores de STA, STI, STJ respectivamente. Os bits de TRANSMISSÃO e "VAL DE" são atualizados conforme a variável STT.

3) CÓDIGOS: TPI

4) FUNÇÃO:

Transferência dos parâmetros associados com a execução de um primitivo.

5) CHAMADA:

PPG TPI

6) PARÂMETROS DE ENTRADA:

Nenhum

7) PARÂMETROS DE SAÍDA:

ACUMULADOR

8) VARIÁVEIS LOCAIS:

Nenhuma

9) VARIÁVEIS GLOBAIS:

EP1, EPJ, NDT

10) ROTINAS UTILIZADAS:

Nenhuma

11) DESCRIÇÃO:

A Variável NDT é utilizada como um índice para a sequência de parâmetros de um primitivo, sequência esta designada pelo endereço contido em EP1, EPJ.

A cada chamada de TPI, o ACUMULADOR é carregado com o parâmetro indicado por NDT utilizando-se o endereçamento indireto pós-indexado. Finalmente o valor de NDT é incrementado.

a) CÓDIGO: API

b) FUNÇÃO:

Armazenar parâmetros na sequência de chamada de um primitivo após a execução deste.

c) CHAMADA:

PUG API

d) PARÂMETROS DE ENTRADA:

ACUMELADOR

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

EPI, EPI, NDT

h) ROTINAS UTILIZADAS:

Nenhuma

i) DESCRIÇÃO:

A sequência de parâmetros de um primitivo é apontada pela variável EPI, EPI, NDT indica a posição do parâmetro nesta sequência. Utilizando o endereçamento indireto pós-indexado, o valor do ACUMELADOR é armazenado nesta posição indicada por EPI, EPI e NDT. A variável NDT é incrementada.



a) CÓDIGO: PXL

b) Função:

Percorrer a lista de processos TDS.

c) CHAMADA:

PUG PXL

d) PARÂMETROS DE ENTRADA:

ACUMULADOR: identificação do processo a partir do qual a lista deve ser percorrida.

e) PARÂMETROS DE SAÍDA

ACUMULADOR: identificação do próximo processo na lista TDS.

ÍNDICE: identificação do processo de partida

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

TDS

h) ROTINAS UTILIZADAS:

Nenhuma

i) DESCRIÇÃO:

O valor contido em TDS na posição indicada pelo ACUMULADOR é separado e a parte correspondente ao encaдрamento da lista (9 bits menos significativos) é devolvida no ACUMULADOR.

a) CÓDIGOS: PON

b) FUNÇÃO:

Pesquisar na tabela de identificação dos processos (IDP) a identificação de um processo dado o seu código.

c) CHAMADA:

PHG PON

d) PARÂMETROS DE ENTRADA:

ACUMULADOR: código do processo

e) PARÂMETROS DE SAÍDA:

ACUMULADOR: identificação do processo. Se o processo não existir o ACUMULADOR é devolvido com o valor zero.

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

TDS, TDN

h) ROTINAS UTILIZADAS:

PXL

i) DESCRIÇÃO:

Os códigos de cada processo, contidos na tabela TDS são comparados com código fornecido. A pesquisa é realizada utilizando-se a rotina PXL. Quando o código for encontrado, a identificação do processo é automaticamente fornecida por PXL.

a) CÓDIGO: PPI

b) FUNÇÃO:

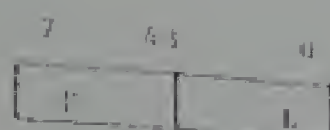
Retirar e inserir elementos em uma lista encadeada

c) CHAMADA:

POG POL

d) PARÂMETROS DE ENTRADA:

ACUMULADOR :



F=0 INSERIR

F=1 RETIRAR

L=0 LISTA DE PROCESSOS (TDS)

L=1 LISTA DE MENSAGENS (TME)

L=2 LISTA DE PROCESSOS SUBORDINADOS (TDS)

ÍNDICE: posição a ser retirada da lista

e) PARÂMETROS DE SAÍDA:

ACUMULADOR:

ACUMULADOR > 0 - posição na lista onde um elemento foi acrescentado

ACUMULADOR = 0 - ERRO

f) VARIÁVEIS LOCAIS:

POI - Variável auxiliar

POF - endereços das listas manipuladas

POQ - endereço da lista sendo manipulada

PIF - posição do primeiro elemento da lista

PIV - posição do primeiro elemento da parte vazia da lista



## 2.1 VARIÁVEIS GLOBAIS:

TOS, TML, TDB

105.

## 2.2 ROTINAS UTILIZADAS:

problema

## 2.3 PRESENTAÇÃO:

As listas TOS, TML e TDB são estruturadas de forma tal que cada uma é constituída por uma tabela contendo 45 posições. Uma posição na lista corresponde a um índice nesta tabela. A posição de índice zero é reservada para o controle da tabela. Em suma, que cada uma das 45 restantes são subdivididas em duas partes. Os quatro bits mais significativos conterão uma informação particular da lista e os quatro bits menos significativos são reservados para o encadeamento da lista.

Cada tabela comporta duas listas, a lista de elementos (LE) e a lista vazia (LV).

A lista de elementos inicia-se em uma posição da tabela indicada pelos quatro bits mais significativos da posição zero, enquanto que os quatro bits menos significativos irão indicar a posição inicial da lista vazia. O encadeamento consiste em, dada uma posição na lista, a posição de próximo elemento desta lista será obtido por meio dos quatro bits menos significativos. De modo similar, se o valor desses quatro bits forem nulos, indicará o término da lista.

A estrutura desta tabela é indicada na Figura 2.6.

Na estrutura desta tabela é indicada na Figura 2.6.



1) ROTINA DE CRIAÇÃO DE PROCESSO

121

a) CÓDIGO: FNI

b) EXTENSÃO:

extensão do processo

c) CHAMADA:

PDG - CMP

d) PARÂMETROS DE ENTRADA:

ACUMULADOR: identificação do próximo processo

e) PARÂMETROS DE SAÍDA:

Nenhuma

f) VARIÁVEIS LOCAIS:

PFX: identificação do processo em execução  
PDI, PDI: endereço do gerador do processo

g) VARIÁVEIS GLOBAIS:

TDL, TDC, NDT, STA, STI, STL, STT

h) ROTINAS UTILIZADAS:

Nenhuma

i) DESCRIÇÃO:

Por meio da variável PFX é obtida a identificação do processo em execução, com essa identificação o endereço do descritor deste processo é lançado nas variáveis PDI e PDI consultando-se as tabelas TDL e TDC, os valores do ACUMULADOR, EXTENSÃO, INÍCIO, CONTADOR DE INSTRUÇÕES, TRANSFERIDO (TI) e "VAL-MEM" (VM) são lidos das variáveis STA, STI, STL, posições 2 e 4 da memória e lançados respectivamente e lançados no descritor por meio do endereço



128.  
Essas variáveis foram atualizadas, por ocasião da execução do processo indicado por R13.

Na sequência, a variável R13 é feita igual ao valor de A. O conteúdo de R13 é transferido para a rotina e da mesma forma interior. O conteúdo de cada DESCRIPTOR é colocado nas variáveis PDI e PDI. Os valores contidos neste DESCRIPTOR são transferidos para as variáveis STA, STE, STI, STR e para as posições 2 e 3 da memória.

A comparação somente será efetuada acionando-se a rotina de verificação de "STATUS" (RST) e pela execução da instrução R13. Por essas duas últimas operações são realizadas externamente à rotina CMP.

A rotina CMP não é utilizada quando um processo for interrompido e acionado novamente ao serviço.

II. FÓRMULA DE MODIFICAÇÃO DO ESTADO DE UM PROCESSO

100.

a) CÂDIGOS: RTT

b) FÓRMULA:

Modificação do estado de um processo

c) COMANDA:

RTT RTT

d) PARÂMETROS DE ENTRADA:

ACUMULADOR: Estado a ser armazenado

ÍNDICE: Identificação do processo

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Nenhuma

g) VARIÁVEIS GLOBAIS:

TDS

h) FÓRMULAS UTILIZADAS:

Nenhuma

i) DESCRIÇÃO:

Por meio do valor de ÍNDICE fornecido, a correspondente posição na tabela TDS é atualizada de acordo com o valor armazenado no ACUMULADOR.

# 15 - ROTINA DE IMPLEMENTAÇÃO DOS PRIMITIVOS

136.

a) CÓDIGO: PRM

b) FUNÇÃO:

Execução dos primitivos

c) CHAMADA:

PBC	PRT	op
PBC	LRI	
PBC	ARI	
PBC	LES	
PBC	ARS	
PBC	LFS	
PBC	CRP	
PBC	ARP	
PBC	LMS	
PBC	AMS	
PBC	AMG	
PBC	PSM	
PBC	RMP	
PBC	LDP	
PBC	TRP	

d) PARÂMETROS DE ENTRADA:

Nenhum

e) PARÂMETROS DE SAÍDA:

Nenhum

f) VARIÁVEIS LOCAIS:

Definidas para cada primitivo

g) VARIÁVEIS GLOBAIS:

RBI, PEX, TDS, FDL, TDC, TDM, TFC, TFB, TD

h) ROTINAS UTILIZADAS:

POL, PON, ATP, CSP, RTE, TPI, API



Cada primitivo é processado por uma subrotina pertencente ao PRM. Os parâmetros associados a cada primitivo são transferidos por meio das rotinas TPI e API. Esses parâmetros são transferidos em uma sequência que acompanha a instrução de geração da interrupção cíntrica e serão designados pelos rótulos  $p_1, p_2, \dots, p_n$  e conterão os valores  $x_1, x_2, \dots, x_n$ .

```

CNP
DEFC      N
P1 DEFC  X1
P2 DEFC  X2
P3 DEFC  X3
:
:
Pn DEFC  Xn

```

#### SUBROTINA PRT

$X_1$  é enviado à interface do RTR por meio de uma instrução "SAI".

#### SUBROTINA LRI

As variáveis RIT, RIA e RIB são colocadas nas posições  $p_1, p_2$  e  $p_3$  respectivamente.

#### SUBROTINA ARI

Os valores  $X_1, X_2$  e  $X_3$  são colocados nas variáveis RIT, RIA e RIB respectivamente.

#### SUBROTINA AES

Uma linha de índice  $X_1$  das tabelas TEL, TER e TEC ("TABELAS CONCENTRADAS") são atualizadas com os valores  $X_2, X_3, X_4$  respectivamente.

# SUBROTINA IES

132.

A interface designada por  $X_1$  é acionada por meio das instruções de Entrada/Saída conforme a operação indicada na tabela IEE.

# SUBROTINA LES

O conteúdo da linha especificada por  $X_1$  da tabela IEE é transferido para  $p_2$ .

# SUBROTINA TRP

As instruções de testes de pedidos de interrupção recebidas pela rotina PDI e ainda os endereços de desvio que acompanham estes testes são lidas de acordo com as posições indicadas pelo parâmetro  $X_1$ .

# SUBROTINA CRP

Por meio da rotina PQN, o código do processo a ser criado ( $X_1$ ) é verificado. Caso já exista um processo com esse código, a rotina ATP é acionada.

Através da rotina PQL é pedida a inclusão de um novo elemento na tabela TDS. Obtida a posição do novo elemento, as tabelas TDN, TDL e TDC são atualizadas nessa posição com os valores de  $p_1$ ,  $p_2$  e  $p_3$  respectivamente. A posição da tabela TDS correspondente ao Estado do novo processo é preenchida com  $(20)_{16}$  correspondendo ao Estado PARADO.

Se inicialmente a tabela TDS já estiver totalmente preenchida a rotina ATP será acionada.

# SUBROTINA AFP

A posição na tabela TDS do processo designado por  $X_1$  é obtida através da rotina PQN. Os quatro bits mais significativos dessa posição de TDS são preenchidos com o valor de  $X_2$ .

# SUBROTINA LDP

Os valores das tabelas TDL e TDC na posição indicada

131.  
pelo código do processo ( $X_1$ ) são transferidos para  $P_2$  e  $P_3$  respectivamente. A posição na tabela é obtida por meio da rotina PQB.

#### SUBROTINA ROP

Como nas demais subrotinas, a posição nas tabelas de descrição dos processos é obtida por meio da rotina PQB, conforme o parâmetro  $X_1$ . Representando-se por  $i$  essa posição, os conteúdos dessas tabelas serão designados por  $TDS(i)$ ,  $TMI(i)$ ,  $TDC(i)$  e  $TDE(i)$ .

Para a remoção dos processos é utilizada a tabela TDR, onde serão assinalados os processos que devem ser removidos, uma vez que a remoção de um processo acarreta a remoção de seus subordinados.

A remoção de cada processo é realizada acionando-se a rotina PQL, com o fornecimento da posição do processo na tabela TDR.

A pesquisa de todos os processos a serem removidos é realizada por meio da tabela de processos subordinados TDB, uma vez que esta tabela contém, para cada processo, uma lista encadeada com a indicação de seus subordinados.

Por meio dos quatro bits mais significativos de  $TDB(i)$  a lista TDB é percorrida e para cada processo  $j$  desta lista é feito  $TDB(j) = 0$ .

Finalmente, a lista TDS é percorrida e para todo processo no qual  $TDB(k) \neq 0$  é acionado a rotina PQL com o parâmetro  $k$ .

#### SUBROTINA RMB

As subrotinas que processam as MENSAGENS trocadas pelos processos utilizam as tabelas TMD, TML e TML descritas no capítulo 2.

O processo REPETENTE, obtido através de PEX, e BESTIÁRIO por meio de  $X_1$  e rotina PQB.

Em seguida, uma posição na tabela TML é obtida através da rotina PQL e  $TMD(i)$  é feito igual a NR enquanto que  $TML(i)$  e  $TML(i)$  receberão os valores  $X_2$  e  $X_3$  respectivamente.



114.  
Caso não exista posição para a tabela TML, a rotina  
ATF é acionada.

#### SUBROTINA AMS

Uma variável  $NR$  é obtida através da variável  $PR$  e do  
parâmetro  $X_1$  (rotina PUX).

O valor de  $NR$  é pesquisado na tabela TML. Se for encon-  
trados as posições correspondentes das tabelas TML e TML são acio-  
nadas em  $P_2$  e  $P_3$ . Caso contrário é acionada a rotina ATF e STA,  
a fim de colocar o processo indicada por  $PR$  no estado de [OPERA].

#### SUBROTINA AUC

Utiliza a mesma variável  $NR$  da subrotina AMS, exceto  
que a parte correspondente à identificação do processo DESTINATÁRIO  
é feita igual a zero. A tabela TML é pesquisada somente quanto à  
identificação do processo DESTINATÁRIO. O restante da subrotina é  
idêntica à subrotina AMS.

#### SUBROTINA IEM

É idêntica à subrotina AUC, exceto no caso em que  $NR$   
não é encontrado na tabela TML. Se isso ocorrer os parâmetros  $P_2$   
e  $P_3$  são feitos iguais a -1.

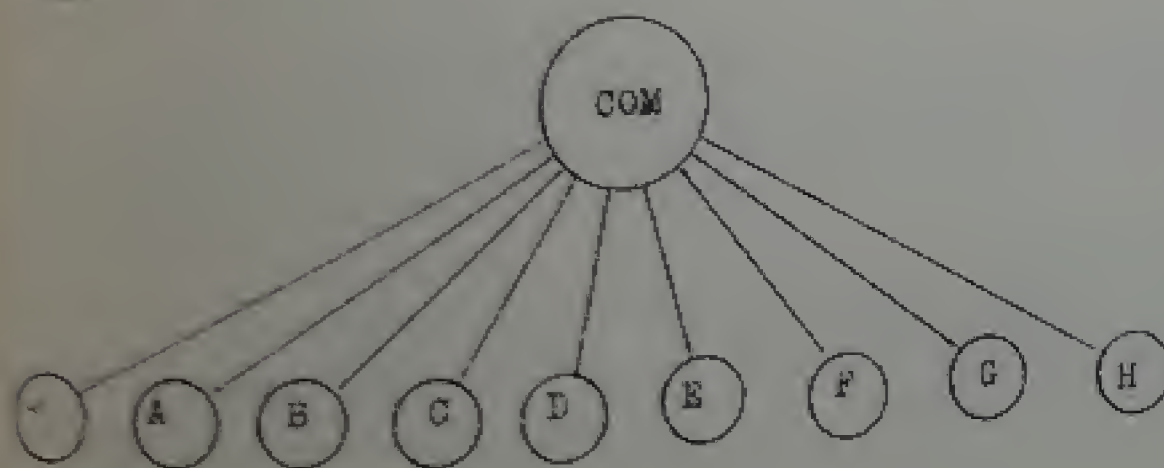
APÉNDICE 2

Um conjunto de 8 processos devem utilizar um recurso comum disponível no sistema. Este recurso, representado por uma impressora de linhas, é atribuído sequencialmente a cada um dos processos mediante uma requisição enviada a um outro processo ou

Aos processos usuários, denominados por A, B, C, D, E, F, G, H, são atribuídas prioridades em relação à utilização da impressora, de tal modo que o processo supervisor S, recebendo uma requisição, deve colocar o processo requisitante em uma "fila" de acordo com a prioridade do mesmo. A atribuição da P.C.P. a um dos processos é suposta realizada em um esquema "round robin", não existindo prioridades em relação à utilização da P.C.P.

## 2. - O SIMULADOR

Um possível sistema que possa realizar as tarefas propostas anteriormente foi organizado de acordo com o esquema abaixo:





O processo S mantém uma "fila", realizada por meio de uma tabela TPR constituída por 8 posições.

Cada posição de TPR está associada a um dos processos, de tal modo que a posição de índice 7 corresponde ao processo H, sendo esta posição a de maior prioridade.

O conteúdo de cada posição de TPR indica se o processo correspondente deseja ou não utilizar a impressora. No primeiro caso  $TPR(i) \neq 0$  e no segundo caso  $TPR(i) = 0$ , sendo mantida a seguinte correspondência:

POSICÃO i	PROCESSO
0	A
1	B
2	C
3	D
4	E
5	F
6	G
7	H

Inicialmente o processo S examina a tabela TPR, sempre a partir da posição 7, verificando qual a primeira posição  $i$  na qual  $TPR(i) \neq 0$ .

Quando esta situação ocorrer,  $TPR(i)$  é feita igual a zero, o número do intervalo de tempo ("time slice") é lido e uma mensagem é enviada ao processo PLS, ordenando que seja impressa uma linha.

Esta linha conterá o nome do processo, o valor do contador correspondente à posição 1 na tabela CNT e o número do intervalo de tempo lido anteriormente.

Em seguida, uma MESSAGE é enviada ao processo correspondente a 1.

Novamente, a tabela TPR é consultada e um novo ciclo é realizado, conforme indicado na figura A2-1.

## 2.2. - OS PROCESSOS A, B, C, D, E, F, G e H

Uma característica desses processos é que eles compartilham um mesmo programa. Este programa, denominado por P, não altera o valor do registrador ÍNDICE (1) de modo que quando em execução o valor deste índice 1 indicará qual dos processos que o utiliza.

Assim, os DESCRITORES dos processos A até H conterão valores fixos para o ÍNDICE e indicarão através do CONTADOR DE INSTRUÇÕES (CI) uma mesma área de códigos.

O comportamento do programa P refletirá portanto o comportamento de qualquer um dos processos A até H.

A área de dados associada a este programa é constituída por três tabelas TPR, CNT e PX contendo 8 posições cada uma. Essas tabelas são utilizadas sempre por meio de instruções "indexadas" de tal forma que os dados TPR(1), CNT(1) e PX(1) são reservadas ao processo com índice 1. Desse modo, o processo A, por exemplo, utiliza somente os valores TPR(0), CNT(0) e PX(0).

Cada um desses processos deverá somar 1 ao contador CNT(1) e comparar o resultado com o limite PX(1). Se CNT(1) for maior que PX(1) uma requisição da impressora é enviada ao processo.

8. fazendo  $TPR(1)$  diferente de zero. Após esta requisição, uma  $MSG$  do processo B é aguardada. Recebida a  $MSG$  de B o ciclo é repetido. Quando o contador exceder o valor limite ( $PXC1$ ) o processo passará a aguardar uma  $MSG$  do processo de comunicação COM.

Se receber uma  $MSG$  de COM, esta deverá conter um novo valor limite de contagem que será armazenado em  $PX(1)$ . Neste caso o contador é reiniciado com zero e o processo retoma a contagem descrita anteriormente. A figura A2.2 esquematiza esses procedimentos.

### 3. - SIMULAÇÃO

Os programas associados aos processos B, A, E, C, D, I, F, G, e H denominados BSC e P foram escritos na linguagem do "CONTADOR RELOCÁVEL" e processados pelo "CARREGADOR RELOCÁVEL" descritos na referência (2) e (15) obtendo-se uma fita perfurada em "FORMATO ABSOLUTO" com início na posição de endereço  $(F00)_{16}$  da memória. Os descritores dos processos A até H, ficaram localizados nas posições dadas pela tabela abaixo, juntamente com os códigos associados a esses processos.

PROCESSO	ENDEREÇO DO DESCRITOR	CÓDIGO
		5
B	$(100)_{16}$	41
A	$(116)_{16}$	42
B	$(119)_{16}$	43
C	$(114)_{16}$	44
B	$(115)_{16}$	45
E	$(102)_{16}$	46
F	$(109)_{16}$	47
G	$(110)_{16}$	48
H	$(117)_{16}$	



A utilização do Sistema Básico de Controle para esta simulação é demonstrada no final deste apêndice, onde comparecem os comandos enviados ao SBC ao lado de alguns comentários e o resultado obtido.

Esses comandos podem ser agrupados em:

#### a) IMPRESSÃO DO TÍTULO

Na cabeça do documento, o título

#### PROCESSO CONTADOR A INSTANTE

foi integralmente perfurada em uma fita de papel e colocado na 1ª tira de fitas. Por meio do envio de uma MENSAGEM ao processo PPS(2) na forma

PPS, 2, 4, 2, 0F, 20, 50

o conjunto de caracteres do Título foi armazenado em posições a partir do endereço (F20) da memória.

Alguns caracteres de controle da impressora foram acrescentados pelo comando CBT e novamente através de uma MENSAGEM enviada ao processo PPS,

PPS, 2, 4, 64, 0F, 20, 26

obteve-se a impressão deste Título na impressora de linhas.

#### b) CARGA DOS PROGRAMAS SSS e P

A fita perfurada contendo os programas SSI e P foi transferida para a memória pelo processo CPE(3) através do comando de envio de Mensagens.

PPS, 3, 4, 0, 2, 0F, 00

Para a verificação do transporte destes programas, o processo CPE foi colocado no estado de "espera" aguardando uma MENSAGEM

SAGLH de CPR, por meio do comando

AMS, 3

Recebida a Mensagem de CPR, esta foi impressa verificando-se que a transferência foi correta uma vez que a primeira palavra da Mensagem é zero.

#### c) CRIAÇÃO DOS PROCESSOS

Conhecendo-se as posições dos Descritores dos processos S e A até H, estes foram adicionados ao SSC por meio do comando CPR.

#### d) ATIVAÇÃO DOS PROCESSOS

Uma vez que os processos recém criados permanecem no estado "PARADO", estes devem ser ativados mediante um comando de "ALTERA ESTADO" AEP, onde se especifica o código do processo e o novo estado assumido.

O processo S foi mantido no estado "PARADO".

#### e) INICIAÇÃO DOS PROCESSOS A, B, C, D, E, F e H

As tabelas CNT e PX relativas ao programa P foram iniciadas de tal forma que na primeira vez que esses processos as consultam, passem a aguardar uma mensagem do processo CDB que irá conter o número de vezes em que a impressão deverá ser requisitada ao processo S.

Nesta forma a iniciação desses processos corresponde ao envio de Mensagens, pelo processo CDB, contendo este número.

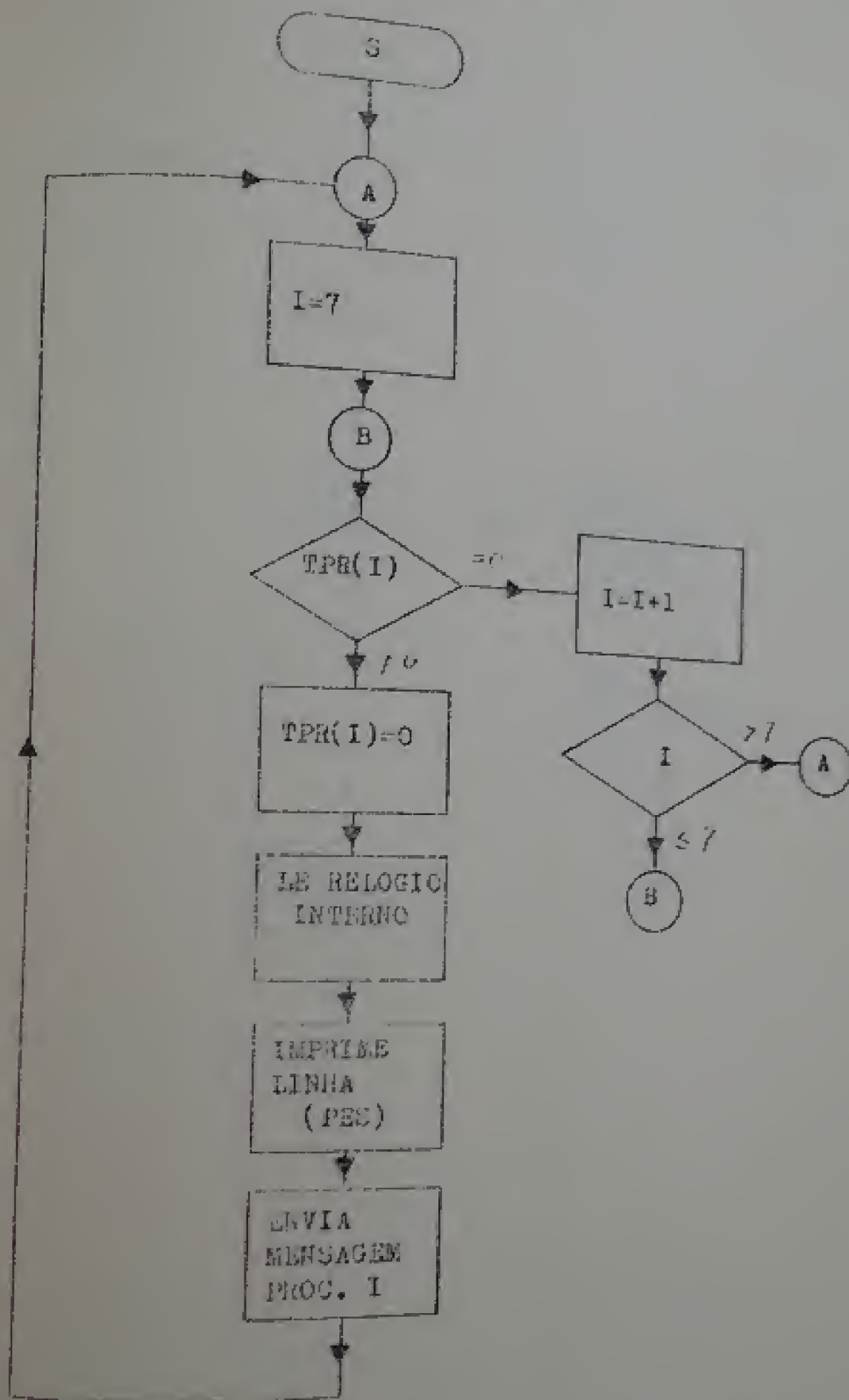
os comandos VHS foram utilizados especificando o código do processo e o número escolhido.

#### f) ATIVAÇÃO DO PROCESSO S

Após a iniciação dos processos A até H, passaram a requisitar a impressora. Entretanto nada foi impresso até que o processo S fosse ativado.

O número do intervalo de tempo foi iniciado com zero por meio do comando AR1,0,0,0 e em seguida o processo S foi ativado pelo comando AEP,5,80 iniciando a utilização da impressora.





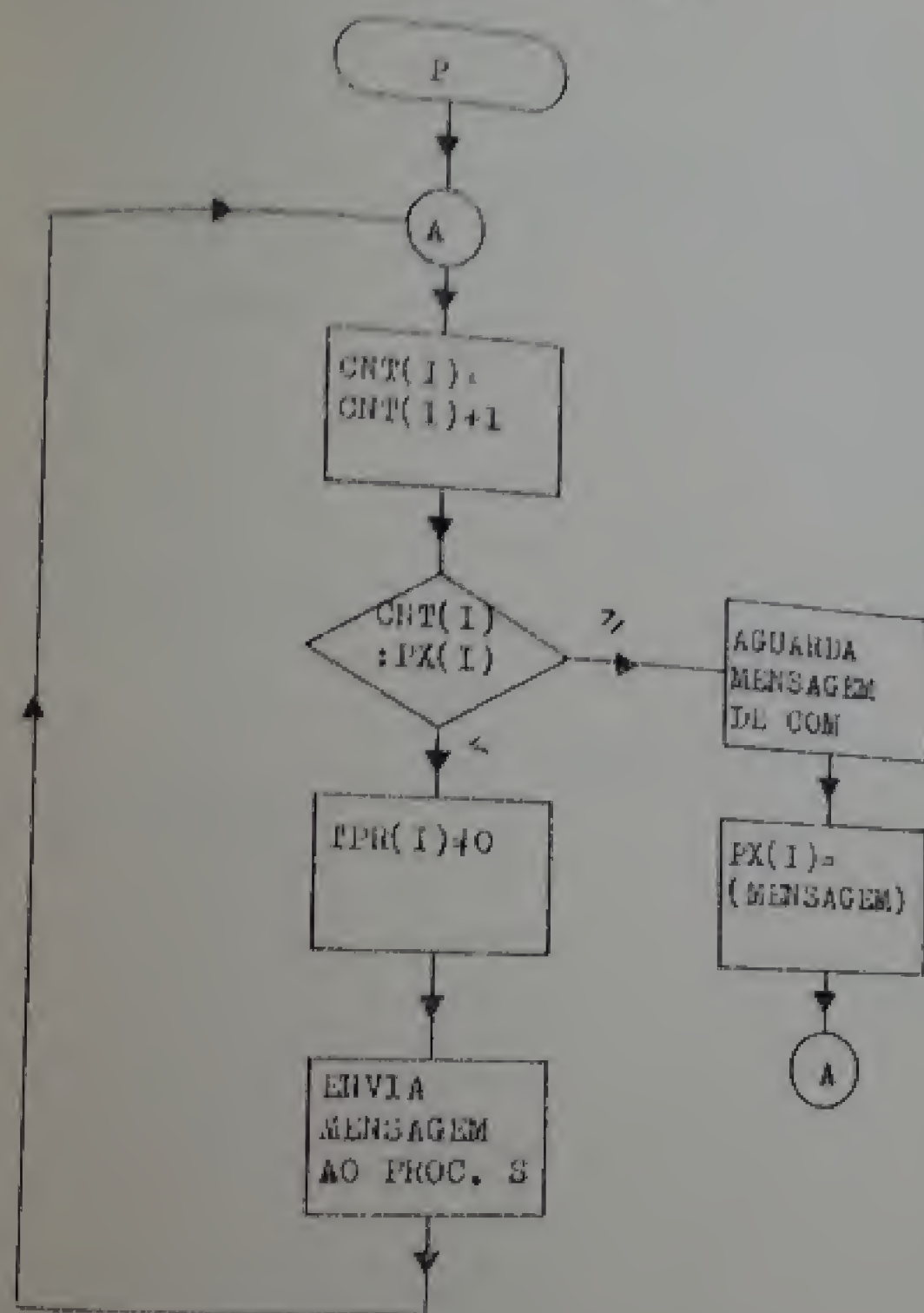


FIG. - A2.2

```

?
C: LECTURA DO TITULO
?
ENS. 2, 4, 2, BF, 20, 50
*
C: CARACTERES DE CONTROLE DO TITULO
?
EDT, E, /F20, 84
*
?
EDT, E, /F45, 82
*
?
C: IMPRIMIR TITULO
?
ENS. 2, 4, 64, BF, 20, 26
*
C: CARGA DOS PROGRAMAS 955 E P EM /E00
?
ENS. 3, 4, 0, 2, 0E, 00
*
?
C: VERIFICAR A CARGA
?
AMS, 3
00 00 00 00 02 00 0F 00 03 22
?
C: CRIAR OS PROCESSOS 5 E 41 ATE 40
?
CRP, 5, /E00
*
?
CRP, 41, /EE6
*
?
CRP, 42, /EED
*
?
CRP, 43, /EF4
*
?
CRP, 44, /EFB
*
?
CRP, 45, /F02
*
?
CRP, 46, /F09
*
?
CRP, 47, /F10
*
?
CRP, 48, /F17
*
?

```

```

C: ATIVAR OS PROCESSOS 41 ATE 40
?
REP, 41, 80
*
?
REP, 42, 80
*
?
REP, 43, 80
*
?
REP, 44, 80
*
?
REP, 45, 80
*
?
REP, 46, 80
*
?
REP, 47, 80
*
?
REP, 48, 80
*
?
C: INICIAR PROCESSOS 41 ATE 40
?
ENS, 41, 3
*
?
ENS, 42, 4
*
?
ENS, 43, 5
*
?
ENS, 44, 6
*
?
ENS, 45, 7
*
?
ENS, 46, 8
*
?
ENS, 47, 9
*
?
ENS, 48, 9
*
?
C: INICIAR RELOGIO INTERNO
?
AR1, 0, 0, 0
*
?
C: ATIVAR PROCESSO 5
?
REP, 5, 80
*
?

```

PROCESSO CONSIDERAR \* ENS

```

H 01 * 00004C
C 01 * 00004E
H 02 * 000052
C 02 * 000054
H 03 * 000058
C 03 * 00005E
H 04 * 000064
C 04 * 00006D
H 05 * 000064
C 05 * 00006E
H 06 * 000064
C 06 * 000064
H 07 * 000070
C 07 * 000072
H 08 * 000076
C 08 * 000078
F 01 * 00007C
E 01 * 00007E
F 02 * 000082
E 02 * 000084
F 03 * 000088
E 03 * 00008E
F 04 * 000092
E 04 * 000094
F 05 * 000098
E 05 * 00009B
F 06 * 00009A
E 06 * 00009E
F 07 * 0000A0
E 07 * 0000A4
C 01 * 0000A4
C 01 * 0000A6
C 02 * 0000AA
C 02 * 0000AC
D 03 * 0000B0
C 03 * 0000B2
E 04 * 0000B4
C 04 * 0000B8
E 05 * 0000C0
E 01 * 0000C0
E 02 * 0000C2
E 02 * 0000C4
E 02 * 0000C8
E 02 * 0000C0

```



R E F E R Ê N C I A S

- (1) FREGNI, EDSON  
"Projeto Lógico da Unidade de Controle de um Minicomputador"  
Dissertação de Mestrado - EPUSP, 1977
- (2) NETO, JOAO JOSÉ  
"Aspectos do Projeto de Software de um Minicomputador"  
Dissertação de Mestrado - EPUSP, 1975
- (3) HANSEN, BRINCH  
"Operating Systems Principles"  
Prentice-Hall, INC. - 1971
- (4) FARWELL, RICHARD  
"Operating Systems: The Key to Minicomputer Systems"  
Data General Corporation - 1973
- (5) DENNING, PETER J.  
"Third Generation Computer Systems"  
Computing Surveys, vol. 3, nº 4, December - 1971
- (6) HEWLETT-PACKARD  
"Real Time Software"  
September - 1969
- (7) BILLS, DAVID L.  
"Executive Systems and Software Development for  
Minicomputers"  
Proceedings of the IEEE, vol. 61, nº 11, November - 1973
- (8) LANGDON JR., GLEN GEORGE e FREGNI, EDSON  
"Projeto de Computadores Digitais"  
Edgard Blücher, Editora da B.S.P. - 1974
- (9) TACHIBANA, MARIO  
"Carregador Relocável para o Computador PATO FEIO"  
Publicação Interna do Lab. de Sistemas Digitais - 1974.

- (10) BOROVAN, JOHN J. e MADNICK S.  
"Operating Systems"  
McGraw-Hill - 1975
- (11) KOVACH, STEPHAN  
"Projeto de um Sistema de Entrada e Saída para  
um Minicomputador"  
Dissertação de Mestrado - 1975.
- (12) MINICOMPUTADOR G-10  
"Manual de Arquitetura - G-10-M01"  
Digibrás - agosto 1975
- (13) HEWLETT-PACKARD  
"Basic Control System"  
February - 1968
- (14) SCHOEFFLER, JAMES D. e BRONNER LEE R.  
"Data Management Software for Mini Production Monitoring  
and Control Systems"  
Proceedings of the IEEE, vol. 61, nº 11, november - 1973
- (15) MINICOMPUTADOR G-10  
"Manual de Operação G-10-M05"  
Digibrás - agosto 1975
- (16) MINICOMPUTADOR G-10  
"Manual do Montador - G-10-M06"  
Digibrás - agosto 1975
- (17) LIR, CHARLIE  
"Carregador Retecável para o PATIBO FEIO: Versão  
em 1 Passo"  
Publicação interna do Lab. de Sistemas Digitais - 1976
- (18) MARTINS JR, MOACYR  
"Projeto de uma Interface de Controle para Memória  
Monolítica: Recurso para Ampliação da Memória Principal  
em Minicomputador".  
Dissertação de Mestrado - a ser publicada.



FD-147

AUTOR Rego, Antônio José de

TÍTULO

Rego

Nº

R

FD - 147

Nº

Rego, Antônio José de

Software de um microcom-  
putador sistema básico de pro-  
gramação.

ESCOLA POLITÉCNICA - USP